Self-paced Consensus Clustering with Bipartite Graph

Peng Zhou[1,2] , **Liang Du**[3] , **Xuejun Li**[1] *

[1]School of Computer Science and Technology, Anhui University
[2]State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences
[3]School of Computer and Information Technology, Shanxi University
zhoupeng@ahu.edu.cn, duliang@sxu.edu.cn, xjli@ahu.edu.cn

Abstract

Consensus clustering provides a framework to ensemble multiple clustering results to obtain a consensus and robust result. Most existing consensus clustering methods usually apply all data to ensemble learning, whereas ignoring the side effects caused by some difficult or unreliable instances. To tackle this problem, we propose a novel self-paced consensus clustering method to gradually involve instances from more reliable to less reliable ones into the ensemble learning. We first construct an initial bipartite graph from the multiple base clustering results, where the nodes represent the instances and clusters and the edges indicate that an instance belongs to a cluster. Then, we learn a structured bipartite graph from the initial one by self-paced learning, i.e., we automatically decide the reliability of each edge and involves the edges into graph learning in order of their reliability. At last, we obtain the final consensus clustering result from the learned bipartite graph. The extensive experimental results demonstrate the effectiveness and superiority of the proposed method.

1 Introduction

Clustering is a fundamental problem in unsupervised learning and attracts many attentions in past decades. However, according to [Wang *et al.*, 2009a], traditional clustering methods suffer from the stable and robust problems. To address these problems, consensus clustering is proposed.

Consensus clustering provides an elegant framework for integrating multiple weak base clusterings to generate a consensus clustering result [Topchy *et al.*, 2004]. In recent years, many consensus clustering methods have been proposed [Strehl and Ghosh, 2003; Zhou and Tang, 2006; Zhou *et al.*, 2015a; Liu *et al.*, 2018]. For example, [Strehl and Ghosh, 2003] and [Topchy *et al.*, 2003] proposed information theoretic based consensus clustering methods; [Zhou and Tang, 2006] proposed an alignment method to combine multiple k-means clustering results; [Liu *et al.*, 2015] provided a spectral clustering based ensemble method; [Wang *et al.*, 2009b] and [Huang *et al.*, 2016a] introduced probabilistic graphical model into consensus clustering. Besides these works which ensembled all base clustering results, some works tried to select some informative and non-redundant base clustering results for ensemble. For example, [Azimi and Fern, 2009] proposed an adaptive clustering ensemble selection method; [Zhao *et al.*, 2017] applied internal validity indices to select based clustering results.

Note that, these methods always apply *all* instances for ensemble learning. However, since the base results may not be fully reliable, it is inappropriate to always use all data for ensemble. Intuitively, some instances are unreliable for clustering or even outliers, which lead to the poor performance of the base clusterings. At the beginning of learning, these unreliable instances may mislead the model, because the early model may not have the ability to handle these unreliable ones.

To address this issue, we propose a novel Self-paced Consensus Clustering with Bipartite Graph (SCCBG), which involves instances from more reliable to less reliable ones into the ensemble learning. The basic idea is that, we train the model with the easier or more reliable instances in the process of ensemble, until it is strong enough to handle the difficult ones. Firstly, we construct an initial bipartite graph from base clustering results, where a node represents an instance or a cluster and an edge indicates that an instance belongs to a cluster. Then, from it, we learn a structured bipartite graph, which contains exact c components, where c is the number of clusters. As introduced before, since the base clustering results are imperfect, some edges in the graph are also unreliable. Therefore, we apply the self-paced learning framework to learn the structured graph, i.e., we automatically decide the reliability of each edge and use the edges for learning in order of their reliability. On the one hand, the reliable edges are helpful to the graph learning; and on the other hand, with the process of graph learning, the edges become more and more reliable. By introducing a carefully designed regularized term to characterize the reliability, we seamlessly integrate the reliability evaluating and graph learning into a unified self-paced learning framework. At last, we obtain the final clustering result from the learned structured graph by finding its connective components. The extensive experiments show that our methods often outperform the state-of-the-art consensus clustering methods.

*Corresponding Author

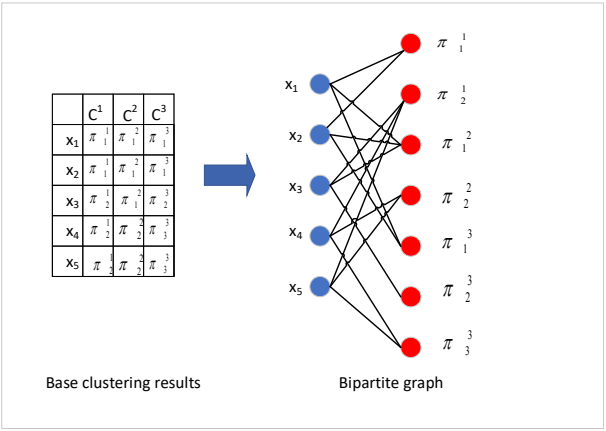


Figure 1: An illustration of constructing the bipartite graph.

2 Preliminaries

Throughout this paper, we use boldface uppercase and lowercase letters to denote matrices and vectors, respectively. For a matrix $\mathbf{M}$, we use $\mathbf{M}_{i\cdot}$ and $\mathbf{M}_{\cdot i}$ to denote the i-th row and column of $\mathbf{M}$, respectively. We denote the (i,j)-th element in $\mathbf{M}$ as M_{ij}.

2.1 Consensus Clustering

Let $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ be a data set with n instances. Given a set of m base clusterings $\mathcal{C} = \{\mathcal{C}^1, \cdots, \mathcal{C}^m\}$ of $\mathcal{X}$, each clustering $\mathcal{C}^i$ consists of a set of clusters $\{\pi_1^i, \cdots, \pi_{k_i}^i\}$, where k_i is the number of clusters in $\mathcal{C}^i$ and $\mathcal{X} = \bigcup_{j=1}^{k_i} \pi_j^i$. Consensus clustering aims to learn a consensus partition of $\mathcal{X}$ from the m base clusterings $\mathcal{C} = \{\mathcal{C}^1, \cdots, \mathcal{C}^m\}$ [Strehl and Ghosh, 2003; Topchy *et al.*, 2003; Topchy *et al.*, 2004].

To learn the consensus partition, in this paper, we construct a bipartite graph $\mathcal{G} = \{\mathcal{V}^1, \mathcal{V}^2, \mathcal{E}\}$ from $\mathcal{C} = \{\mathcal{C}^1, \cdots, \mathcal{C}^m\}$. 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_j^i ($i = 1, \cdots, m$, and $j = 1, \cdots, k_i$). $\mathcal{E}$ 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_p^q, then there is an edge between $\mathbf{x}_i$ and π_p^q. Fig. 1 shows an illustration of constructing the bipartite graph. In this example, we have 5 instances $\mathbf{x}_1, \cdots, \mathbf{x}_5$ and 3 base clusterings. For example, in the first clustering $\mathcal{C}^1$, $\mathbf{x}_1$ 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_2^1. The right side of Fig.1 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. After obtaining the bipartite graph $\mathcal{G}$, we aim to learn a partition on $\mathcal{G}$ as the consensus clustering result.

2.2 Self-paced Learning

The basic idea of self-paced learning is to incrementally involve instances into learning, where easy ones are involved first and difficult ones are then involved gradually [Kumar *et al.*, 2010]. More formally, given a data set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is the feature vector of the i-th instance and y_i is its label, we denote $L(g(\mathbf{x}_i, \boldsymbol{\theta}), y_i)$ as the loss function of the i-th instance, where $g(\mathbf{x}_i, \boldsymbol{\theta})$ is the decision function and $\boldsymbol{\theta}$ is the model parameter. According to [Zhang *et al.*, 2017], the self-paced learning introduces a weighted loss term on instances and a general regularized term on the weights as follows:

$$\min_{\mathbf{w}, \boldsymbol{\theta}} \quad \sum_{i=1}^{n} w_i L(g(\mathbf{x}_i, \boldsymbol{\theta}), y_i) + f(w_i, \lambda). \tag{1}$$

where λ is the "age" parameter to control the learning pace and grows with the learning process, and $f(w_i, \lambda)$ is the self-paced regularized term.

In Eq.(1), if we fix the model parameter $\boldsymbol{\theta}$, supposing $w_i^*(\lambda, l_i)$ is the optimum weight of $\mathbf{x}_i$, where $l_i = L(g(\mathbf{x}_i, \boldsymbol{\theta}), y_i)$, then $f(w_i, \lambda)$ should satisfy that $w_i^*(\lambda, l_i)$ is monotonically decreasing with l_i and increasing with λ as suggested in [Jiang *et al.*, 2015; Meng *et al.*, 2017]. Note that, on the one hand, since $w_i^*(\lambda, l_i)$ is decreasing with l_i, easy instances, which has low loss, will have a large weight, which means they will be involved in learning first. On the other hand, $w_i^*(\lambda, l_i)$ is increasing with λ, which means with the learning process (λ grows), more and more instances are involved in learning.

Therefore, self-paced learning optimizes Eq.(1) via alternating minimization. Fixing $\boldsymbol{\theta}$ and solving $\mathbf{w}$ is to learn the weight of each instance and finding the easy ones; solving $\boldsymbol{\theta}$ by fixing $\mathbf{w}$ is to learn the model using the easy instances. Due to its promising performance and the benefit of alleviating the local optimum problem in non-convex optimization [Basu and Christensen, 2013], self-paced learning has been widely used in many scenarios, such as multi-task learning [Li *et al.*, 2017], robust classification [Ren *et al.*, 2017] and subspace learning [Jiang *et al.*, 2018]. In this paper, we will extend it to unsupervised ensemble learning.

3 Self-paced Consensus Clustering with Bipartite Graph

In this section, we introduce the proposed SCCBG method in detail.

3.1 Formulation

Given m base clustering results, we first construct the initial bipartite graph as introduced in Section 2.1. Note that this initial bipartite graph may not have a very clear clustering structure since each base clustering may not be perfect. Taking Fig. 1 as an example, we find that there is only 1 connective component in the graph, i.e., 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 graph $\mathcal{G}'$ which has exact c connective components where c is the number of clusters. Then clustering on $\mathcal{G}'$ is trivial because we just need to put instances in the same connective components into the same cluster.

More formally, we define the adjacent matrix of $\mathcal{G}$ as

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

where $\mathbf{Y} \in \{0, 1\}^{n \times k}$ and $k = \sum_{i=1}^m k_i$ is the total number of clusters in all clustering results. $Y_{ij} = 1$ means there is an

edge linking the i-th instance ($\mathbf{x}_i$) and the j-th cluster (π_j), and $Y_{ij} = 0$ means there is no edge between them.

Similarly, we can define the adjacent matrix of the structured bipartite graph $\mathcal{G}'$ as

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

where $\mathbf{S} \in [0,1]^{n\times k}$. To make $\mathcal{G}'$ preserve $\mathcal{G}$ as well as possible, we should minimize $\|\mathbf{S}-\mathbf{Y}\|_F^2$. 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 can 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 [Nie *et al.*, 2017], 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:

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

As introduced before, since the base clusterings may not be perfect, 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 the self-paced learning framework seamlessly. We involve edges gradually from more reliable edges to less reliable ones. As suggested in [Jiang *et al.*, 2015], we set $f(w_i, \lambda)$ in Eq.(1) as $-\lambda\|\mathbf{W}\|_1$, and obtain:

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

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 Eq.(5). We need to take a closer look at $\mathbf{W}$. Note that, if two cluster π_p and π_q are similar, then for any instance $\mathbf{x}_i$, either $\mathbf{x}_i$ belongs to the 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 characterise 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. $\mathbf{C}$ can be easily obtained by $\mathbf{C} = \mathbf{Y}^T\mathbf{Y}$. 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), then by minimizing Eq.(6), at least one of W_{ip} and W_{iq} should be small. Taking it into our self-paced framework (Eq.(5)), we obtain the final objective function:

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

where γ is a balanced parameter.

3.2 Optimization

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

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

Then we optimize $\mathbf{W}$, $\mathbf{F}$, $\mathbf{S}$ respectively by fixing the other variables.

Optimizing W

When optimizing $\mathbf{W}$, we find that Eq.(8) can be decoupled into n independent subproblems by rows. Considering the i-th subproblem, we have

$$\begin{aligned} \min_{\mathbf{W}_{i\cdot}} \quad & \sum_{p=1}^{k} W_{ip}^2 A_{ip}^2 - \lambda\sum_{p=1}^{k} W_{ip} + \gamma\sum_{p,q=1}^{k} W_{ip}B_{pq}W_{iq} \\ s.t. \quad & 0 \le W_{ij} \le 1. \end{aligned} \tag{9}$$

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

Note that, Eq.(9) is a quadratic programming with bounded constraint and can be solved by standard optimization method, such as trust region reflective algorithm. In our implication, we use *quadprog* function provided in Matlab.

Optimizing F

When optimizing $\mathbf{F}$, we need to solve the following subproblem:

$$\min_{\mathbf{F}^T\mathbf{F}=\mathbf{I}} \quad tr(\mathbf{F}^T\mathbf{L}\mathbf{F}) \tag{10}$$

Eq.(10) can be solved by computing the eigen-decomposition of $\mathbf{L}$. 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 $\mathbf{L}$ is a Laplacian matrix of

a bipartite graph, according to Lemma 1 in [Nie *et al.*, 2017], Eq.(10) 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 then compute the SVD of $\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{S}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ as $\hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{S}\tilde{\mathbf{D}}^{-\frac{1}{2}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ contain the left and right singular vectors respectively, and $\mathbf{\Sigma}$ is a diagonal matrix which contains the singular values. Let $\mathbf{U}' \in \mathbb{R}^{n\times c}$ and $\mathbf{V}' \in \mathbb{R}^{n\times c}$ be the c left and right singular vectors in $\mathbf{U}$ and $\mathbf{V}$ corresponding to the smallest c singular values, respectively. Then, the closed-form solution of Eq.(10) is:

$$\mathbf{F} = \begin{bmatrix} \mathbf{U}' \\ \mathbf{V}' \end{bmatrix} \tag{11}$$

Note 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 $O((n+k)^3)$.

Optimizing S

When optimizing $\mathbf{S}$, Eq.(8) can also be decoupled into n subproblems by rows. Considering the i-th subproblem, we have:

$$\begin{aligned} \min_{\mathbf{S}_{i\cdot}} \quad & \sum_{p=1}^{k} W_{ip}^2 (S_{ip} - Y_{ip})^2 + \gamma \sum_{p,q=1}^{k} E_{pq}(S_{ip} - S_{iq})^2 \\ & + \rho \sum_{p=1}^{k} H_{ip} S_{ip} \\ s.t. \quad & 0 \le S_{ip} \le 1. \end{aligned} \tag{12}$$

where $E_{pq} = C_{pq} W_{ip} W_{iq}$, and $H_{ip} = \frac{1}{2} \left\| \frac{\mathbf{F}_{i\cdot}}{\sqrt{d_i}} - \frac{\mathbf{F}_{p\cdot}}{\sqrt{d_p}} \right\|_2^2$, and $d_i = \sum_{j=1}^{k} S_{ij}$ and $d_p = \sum_{j=1}^{k} S_{pj}$.

Similar to Eq.(9), Eq.(12) is also a quadratic programming with bounded constraint and can be solved similarly.

Algorithm 1 summarizes the whole process of SCCBG.

Algorithm 1 SCCBG

Input: m base clustering results, number of clusters c, parameter γ.
Output: Consensus clustering results

```
Construct the initial bipartite graph from m based clustering and obtain Y, and initialize the age parameter λ = 0.5, S = Y.
Compute the cluster similarity matrix C = Y^T Y.
while not converge do
    Update W by solving Eq.(9).
    Update F by solving Eq.(10).
    Update S by solving Eq.(12).
    Update the age parameter by λ = λ * 2.
end while
Obtain the final bipartite graph G' from S.
Obtain the final clustering result from the c connective component in G'.
```

	#instances	#features	#classes
ALLAML	72	7129	2
GLIOMA	50	4434	4
K1b	2340	21839	6
Lung	203	3312	5
Medical	706	1449	17
Tr41	878	7454	10
Tdt2	10212	36771	96
TOX	171	5748	4

Table 1: Description of the data sets.

3.3 Discussion

Firstly, we analyze the space and time complexity of our method. Since the graph we used is a bipartite graph, the space complexity of our method is $O(nk)$.

For the time complexity, we analyze it step by step. Firstly, computing $\mathbf{C}$ needs $O(nk^2)$ time. Then, in each iteration, when updating $\mathbf{W}$, we need to solve n quadratic programming problem and each problem involves k variables. Note that, in practice, we set γ a small value to make sure the quadratic programming problem is convex. Therefore, each subproblem costs $O(k^3)$ time and updating $\mathbf{W}$ costs $O(nk^3)$ time. Updating $\mathbf{F}$ needs $O(nk^2)$ as introduced in previous subsection. Updating $\mathbf{S}$ also needs to solve n quadratic programming problem and each problem involves k variables. The time complexity is also $O(nk^3)$. Supposing the number of iterations is l, the whole time complexity is $O(lnk^3)$.

Last but not the least, we discuss the robust consensus clustering, which is very related to our self-paced ensemble. Robust consensus clustering extracts the noises from data or base results and recovers the clean results for ensemble. For example, [Tao *et al.*, 2016; Tao *et al.*, 2019] proposed robust consensus clustering methods based on spectral clustering; [Huang *et al.*, 2016b] used probability trajectories to robust consensus clustering; [Wang *et al.*, 2019] provided an ensemble method on incomplete data. These methods only focus on the noises or outliers without distinguishing between uncontaminated instances. However, in our method, the contaminated instances can be regarded as the most unreliable ones, and in addition, the uncontaminated instances can also be handled in order of reliability. Therefore, our method provides a more sophisticated framework to handle instances. Moreover, in our self-paced framework, the reliability of edges is changing in the process of learning. With the growth of λ, $\mathbf{W}$ will be increasingly large until it reaches 1, which means the edges become increasingly more reliable with learning.

4 Experiments

In this section, we conduct the extensive experiments by comparing our SCCBG with several state-of-the-art consensus clustering methods on benchmark data sets.

Methods	ALLAML	GLIOMA	K1b	Lung	Medical	Tr41	Tdt2	Tox
KM	0.6545	0.4239	0.6726	0.7114	0.3996	0.5626	0.4104	0.4229
KM-best	0.7292	0.4880	0.8559	0.8675	0.4707	0.6946	0.4460	0.4825
CSPA	0.6583	0.4100	0.4531	0.4138	0.3500	0.5213	0.2850	0.4246
HGPA	0.5444	0.4180	0.5326	0.5025	0.2950	0.4894	0.2959	0.3854
MCLA	0.6722	0.4000	0.7383	0.7084	0.4017	0.5698	0.4000	0.4152
NMFC	0.6722	0.4140	0.5860	0.6764	0.3789	0.6323	0.3716	0.4269
BCE	0.6708	0.4280	0.6345	0.6700	0.3965	0.6205	0.1806	0.4140
RCE	0.6708	0.4260	0.6887	0.7143	0.3851	0.6391	-	0.4105
MEC	0.6056	0.3940	0.8190	0.7379	0.3627	0.6559	-	**0.4304**
LWEA	0.6736	0.4320	0.8279	0.7458	0.4208	0.6719	0.5744	0.4234
LWGP	0.6750	0.4320	0.7172	0.6498	0.4047	0.6483	0.4288	0.4193
RSEC	0.5917	0.4180	0.8409	0.8217	0.3490	0.6367	0.4222	0.4041
DREC	**0.6819**	0.4280	0.6462	0.6379	0.3926	0.6243	0.3684	0.4205
SCCBG-W	0.6681	0.4080	0.8405	0.8094	0.3980	0.6136	0.5011	0.4053
SCCBG	**0.6861**	**0.4500**	**0.8663**	**0.8961**	**0.4592**	**0.6973**	**0.7164**	**0.4339**

Table 2: ACC results on all the data sets

4.1 Data Sets

We use 8 data sets, including ALLAML[1], GLIOMA[1], K1b [Zhao and Karypis, 2004], Lung[1], Medical [Zhou *et al.*, 2015b], Tdt2[2], Tr41 [Zhao and Karypis, 2004], and TOX[1]. The details of these data sets are summarized in Table 1.

4.2 Experimental Setup

Following the experimental setup in [Wang *et al.*, 2009b; Zhou *et al.*, 2015b], we use k-means to generate the base clusterings. In more detail, we run k-means 200 times with different initializations to obtain 200 base results. Then we divide them into 10 subsets, with 20 base results in each subset. Next, we apply consensus clustering methods on each subsets, and report the average results on the 10 subsets. We compare our SCCBG with the following methods:

- **KM**, which is the average result of all base clustering.
- **KM-best**, which is the best result of all base results.
- **Cluster-based Similarity Partitioning Algorithm** (**CSPA**) [Strehl and Ghosh, 2003], which signifies a relationship between instances in the same cluster to establish a measure of pairwise similarity for ensemble.
- **HyperGraph Partitioning Algorithm** (**HGPA**) [Strehl and Ghosh, 2003], which integrates base results with a constrained minimum cut objective.
- **Meta-CLustering Algorithm** (**MCLA**)[Strehl and Ghosh, 2003], which transforms the ensemble into a cluster correspondence problem.
- **Nonnegative Matrix Factorization based Consensus clustering** (**NMFC**) [Li and Ding, 2008], which uses NMF to aggregate base results.
- **Bayesian Clustering Ensemble** (**BCE**) [Wang *et al.*, 2009b], which is a Bayesian model for ensemble.
- **Robust Clustering Ensemble** (**RCE**) [Zhou *et al.*, 2015b], which learns a robust consensus result via minimize the KL divergence among each base result.
- **Multi-view Ensemble Clustering** (**MEC**) [Tao *et al.*, 2017], which is a robust multi-view consensus clustering method using low-rank and sparse decomposition to ensemble base clustering and detect the noises.
- **Locally Weighted Evidence Accumulation** (**LWEA**) [Huang *et al.*, 2018], which is a hierarchical agglomerative consensus clustering method based on uncertainty estimation and local weighting strategy.
- **Locally Weighted Graph Partitioning** (**LWGP**) [Huang *et al.*, 2018], which is a graph partition method based on the local weighting strategy.
- **Robust Spectral Ensemble Clustering** (**RSEC**) [Tao *et al.*, 2019], which is a robust clustering ensemble method based on spectral clustering.
- **Dense Representation Ensemble Clustering** (**DREC**) [Zhou *et al.*, 2019], which learns a dense representation for clustering ensemble.
- **SCCBG-W**, which is our ensemble method without self-paced learning. In more detail, we fix the weight matrix $\mathbf{W}$ in SCCBG as 1 and do not update it.

The number of clusters is set to the true number of classes for all data sets and algorithms. Our method adjusts λ automatically as introduced in Algorithm 1. The parameter ρ is also automatically decided. We first initialize $\rho = 1$, and then, if the rank of $\mathbf{L}$ is larger than $n + k - c$, we double it. If its rank is smaller than $n + k - c$, we reduce ρ by half. The only hyper-parameter needed to tune manually is γ. As discussed before, γ should not be too large to make the subproblem convex, thus we tune it in the range $[10^{-5}, 10^{0}]$. We use Accuracy (ACC) and Normalized Mutual Information (NMI) to evaluate the clustering performance. To validate the statistic significance of results, we also calculate the p-value of t-test.

[1]http://featureselection.asu.edu/datasets.php

[2]http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html

Methods	ALLAML	GLIOMA	K1b	Lung	Medical	Tr41	Tdt2	Tox
KM	0.0882	0.1629	0.5493	0.5284	0.4209	0.5843	0.6111	0.1374
KM-best	0.1772	0.2347	0.6853	0.6558	0.4806	0.6713	0.6240	0.2164
CSPA	0.0815	0.1716	0.4071	0.3712	0.3992	0.5919	0.5589	0.1436
HGPA	0.0110	0.1509	0.3917	0.3372	0.3613	0.5084	0.5385	0.1083
MCLA	0.0909	0.1327	0.5944	0.5258	0.4296	0.6044	0.6070	0.1329
NMFC	0.0909	0.1550	0.4995	0.5202	0.4259	0.6512	0.5930	0.1434
BCE	0.0821	0.1658	0.5414	0.4977	0.4499	0.6398	0.0000	0.1370
RCE	0.0899	0.1624	0.6068	0.5248	0.4475	0.6499	-	0.1344
MEC	0.0485	0.1312	0.6818	0.5617	0.4089	**0.6758**	-	0.1313
LWEA	0.0935	0.1686	0.6948	0.5364	0.4185	0.6666	0.7183	0.1236
LWGP	0.0932	0.1682	0.6115	0.4993	0.4266	0.6535	0.6266	0.1333
RSEC	0.0495	0.1544	0.6615	0.6027	0.4036	0.6449	0.5243	0.1184
DREC	0.1006	0.1641	0.5774	0.4647	**0.4510**	0.6514	0.5971	0.1394
SCCBG-W	0.0894	0.1567	0.6888	0.5785	0.3220	0.6039	0.6433	0.1239
SCCBG	**0.1252**	**0.2163**	**0.7262**	**0.6930**	0.3918	**0.6847**	**0.7568**	**0.2131**

Table 3: NMI results on all the data sets

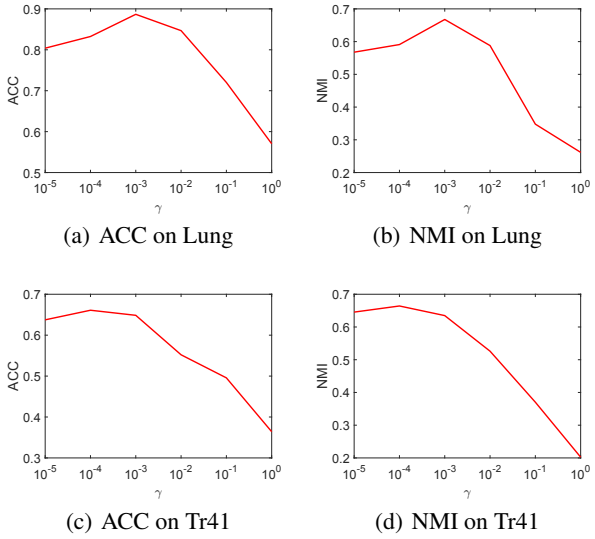
4.3 Experimental Results

Tables 2 and 3 shows the ACC and NMI results of all ensemble methods. The bold fonts means the difference is statistically significant (i.e., the p-value is smaller than 0.05). Note that, due to their high time and space complexity, RCE and MEC run out of memory on the largest data set Tdt2.

From these tables, we find that: (1) many ensemble methods outperform the KM, which indicates the effectiveness of consensus clustering. (2) Compared with other ensemble methods, our algorithm outperforms them on most data sets, which demonstrates its superiority. Even compared with the robust methods (RCE, MEC, and RSEC), ours also performs better, because our self-paced framework can handle data more finely as discussed in Section 3.3. Moreover, to further demonstrate the effectiveness of the self-paced learning framework, we also compare it with SCCBG-W, which is the version without self-paced learning. From the tables, we can see that our method significantly outperforms it, which also demonstrates the necessity of the self-paced learning framework. (3) On most data sets, our method is closed to or even better than KM-best. Note that, ours does not need to perform exhaustive search on the predefined pool of base clusterings, which also well demonstrates the effectiveness of SCCBG.

4.4 Parameter Study

The only hyper-parameter needed to tune manually is γ. As discussed before, γ should not be too large to guarantee the convexity of the subproblems. So we tune it in $[10^{-5}, 10^{0}]$. Fig. 2 shows the results on Lung and Tr41 data sets. We can see that our method works well when γ is in the range $[10^{-5}, 10^{-3}]$. When γ becomes larger, the performance will deteriorate, which is in line with our previous discussion.



(a) ACC on Lung (b) NMI on Lung

(c) ACC on Tr41 (d) NMI on Tr41

Figure 2: Clustering results on Lung and Tr41 with respect to different values of γ.

5 Conclusion

In this paper, we proposed a novel self-paced consensus clustering method with the bipartite graph. We constructed an initial bipartite graph based on the base results. Then we learned a structured graph from it. In the process of graph learning, we adopted the idea of self-paced learning, which automatically decided the reliability of each edge and involves the edges into graph learning in the order of reliability. At last, we obtained the final consensus clustering result by finding the connective components of the learned graph. We conducted extensive experiments on benchmark data sets. Compared with other state-of-the-art consensus clustering methods, our method often performs better than them, which well demonstrates the effectiveness and superiority of the proposed method.

Acknowledgements

This work is supported by the National Natural Science Fund of China grants 61806003, 61976129, and 61972001.

References

[Azimi and Fern, 2009] Javad Azimi and Xiaoli Fern. Adaptive cluster ensemble selection. In *IJCAI*, pages 992–997, 2009.

[Basu and Christensen, 2013] Sumit Basu and Janara Christensen. Teaching classification boundaries to humans. In *AAAI*, pages 109–115, 2013.

[Huang *et al.*, 2016a] Dong Huang, Jianhuang Lai, and Changdong Wang. Ensemble clustering using factor graph. *Pattern Recognition*, 50:131–142, 2016.

[Huang *et al.*, 2016b] Dong Huang, Jianhuang Lai, and Changdong Wang. Robust ensemble clustering using probability trajectories. *IEEE TKDE*, 28(5):1312–1326, 2016.

[Huang *et al.*, 2018] Dong Huang, Changdong Wang, and Jianhuang Lai. Locally weighted ensemble clustering. *IEEE Transactions on Systems, Man, and Cybernetics*, 48(5):1460–1473, 2018.

[Jiang *et al.*, 2015] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. Self-paced curriculum learning. In *AAAI*, pages 2694–2700, 2015.

[Jiang *et al.*, 2018] Yangbangyan Jiang, Zhiyong Yang, Qianqian Xu, Xiaochun Cao, and Qingming Huang. When to learn what: Deep cognitive subspace clustering. In *2018 ACM Multimedia*, pages 718–726. ACM, 2018.

[Kumar *et al.*, 2010] M P Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *NIPS*, pages 1189–1197, 2010.

[Li and Ding, 2008] Tao Li and Chris H Q Ding. Weighted consensus clustering. In *SDM*, pages 798–809, 2008.

[Li *et al.*, 2017] Changsheng Li, Junchi Yan, Fan Wei, Weishan Dong, Qingshan Liu, and Hongyuan Zha. Self-paced multi-task learning. In *AAAI*, pages 2175–2181, 2017.

[Liu *et al.*, 2015] Hongfu Liu, Tongliang Liu, Junjie Wu, Dacheng Tao, and Yun Fu. Spectral ensemble clustering. In *SIGKDD*, pages 715–724, 2015.

[Liu *et al.*, 2018] Xinwang Liu, Xinzhong Zhu, Miaomiao Li, Lei Wang, Chang Tang, Jianping Yin, Dinggang Shen, Huaimin Wang, and Wen Gao. Late fusion incomplete multi-view clustering. *IEEE TPAMI*, pages 2410–2423, 2018.

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

[Nie *et al.*, 2017] Feiping Nie, Xiaoqian Wang, Cheng Deng, and Heng Huang. Learning a structured optimal bipartite graph for co-clustering. In *NIPS*, pages 4129–4138, 2017.

[Ren *et al.*, 2017] Yazhou Ren, Peng Zhao, Yongpan Sheng, Dezhong Yao, and Zenglin Xu. Robust softmax regression for multi-class classification with self-paced learning. In *IJCAI*, pages 2641–2647, 2017.

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

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

[Tao *et al.*, 2017] Zhiqiang Tao, Hongfu Liu, Sheng Li, Zhengming Ding, and Yun Fu. From ensemble clustering to multi-view clustering. In *IJCAI*, pages 2843–2849, 2017.

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

[Topchy *et al.*, 2003] Alexander Topchy, Anil K Jain, and William F Punch. Combining multiple weak clusterings. In *ICDM*, pages 331–338, 2003.

[Topchy *et al.*, 2004] Alexander Topchy, Anil K Jain, and William F Punch. A mixture model for clustering ensembles. In *SDM*, pages 379–390, 2004.

[Wang *et al.*, 2009a] Fei Wang, Xin Wang, and Tao Li. Generalized cluster aggregation. In *IJCAI*, pages 1279–1284, 2009.

[Wang *et al.*, 2009b] Hongjun Wang, Hanhuai Shan, and Arindam Banerjee. Bayesian cluster ensembles. In *SDM*, pages 211–222, 2009.

[Wang *et al.*, 2019] Siwei Wang, Xinwang Liu, En Zhu, Chang Tang, Jiyuan Liu, Jingtao Hu, Jingyuan Xia, and Jianping Yin. Multi-view clustering via late fusion alignment maximization. In *IJCAI*, pages 3778–3784, 2019.

[Zhang *et al.*, 2017] Dingwen Zhang, Deyu Meng, and Junwei Han. Co-saliency detection via a self-paced multiple-instance learning framework. *IEEE TPAMI*, 39(5):865–878, 2017.

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

[Zhao *et al.*, 2017] Xingwang Zhao, Jiye Liang, and Chuangyin Dang. Clustering ensemble selection for categorical data based on internal validity indices. *Pattern Recognition*, 69:150–168, 2017.

[Zhou and Tang, 2006] Zhihua Zhou and Wei Tang. Clusterer ensemble. *KBS*, 19(1):77–83, 2006.

[Zhou *et al.*, 2015a] Peng Zhou, Liang Du, Lei Shi, Hanmo Wang, and Yi-Dong Shen. Recovery of corrupted multiple kernels for clustering. In *IJCAI*, pages 4105–4111, 2015.

[Zhou *et al.*, 2015b] Peng Zhou, Liang Du, Hanmo Wang, Lei Shi, and Yidong Shen. Learning a robust consensus matrix for clustering ensemble via kullback-leibler divergence minimization. In *IJCAI*, pages 4112–4118, 2015.

[Zhou *et al.*, 2019] Jie Zhou, Hongchan Zheng, and Lulu Pan. Ensemble clustering based on dense representation. *Neurocomputing*, pages 66–76, 2019.