Information Fusion 78 (2022) 171–179

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Full length article

Clustering ensemble via structured hypergraph learning

Peng Zhou [a,b,*], Xia Wang [a], Liang Du [c], Xuejun Li [a]

[a] *School of Computer Science and Technology, Anhui University, Hefei 230601, China*
[b] *The State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China*
[c] *School of Computer and Information Technology, Shanxi University, Taiyuan 030006, China*

ARTICLE INFO

Keywords:
Clustering ensemble
Ensemble learning
Hypergraph graph learning

ABSTRACT

Clustering ensemble integrates multiple base clustering results to obtain a consensus result and thus improves the stability and robustness of the single clustering method. Since it is natural to use a hypergraph to represent the multiple base clustering results, where instances are represented by nodes and base clusters are represented by hyperedges, some hypergraph based clustering ensemble methods are proposed. Conventional hypergraph based methods obtain the final consensus result by partitioning a pre-defined static hypergraph. However, since base clusters may be imperfect due to the unreliability of base clustering methods, the pre-defined hypergraph constructed from the base clusters is also unreliable. Therefore, directly obtaining the final clustering result by partitioning the unreliable hypergraph is inappropriate. To tackle this problem, in this paper, we propose a clustering ensemble method via structured hypergraph learning, i.e., instead of being constructed directly, the hypergraph is dynamically learned from base results, which will be more reliable. Moreover, when dynamically learning the hypergraph, we enforce it to have a clear clustering structure, which will be more appropriate for clustering tasks, and thus we do not need to perform any uncertain postprocessing, such as hypergraph partitioning. Extensive experiments show that, our method not only performs better than the conventional hypergraph based ensemble methods, but also outperforms the state-of-the-art clustering ensemble methods.

1. Introduction

Clustering ensemble learns a consensus clustering result from multiple weak base clustering methods [1,2]. Since it can improve the stability and robustness of single clustering methods, it attracts a lot of attention [3–9]. For example, Fern et al. constructed a bipartite graph to represent multiple base clusters and partitioned such bipartite graph for clustering ensemble [3]; Zhou et al. proposed an alignment method to ensemble multiple kmeans clustering results [4].

Among these methods, one kind of the popular methods is the graph based clustering ensemble method. It uses an undirect graph to represent multiple base clusters and learns the final consensus clustering result by partitioning the graph. For example, Mimaroglu et al. constructed a similarity graph to combine multiple clustering results [10]; Zhou et al. proposed a robust clustering ensemble method via multiple graph learning [11]. Although the graph has being demonstrated promising performance in many applications [12,13], the graph structure sometimes may fail to capture the complex high-order correlation between instances. In clustering ensemble tasks, since different base clustering methods may discover the different structures of data, the relationship between instances may be too complex to be characterized by graphs.

To characterize the complex high-order relationship between instances, hypergraph is introduced in the clustering ensemble tasks. Hypergraph contains a node set and a hyperedge set. Different from the edge in the graph, where each edge only links two nodes, the hyperedge can connect any number of nodes, and thus the hypergraph can easily reveal the high-order correlation between instances [14]. Strehl et al. constructed a hypergraph to represent the multiple base clusters, where each instance was represented by a node, and each base cluster was represented by a hyperedge [1]. Then they applied a hypergraph partitioning algorithm to divide the nodes into some clusters. However, since the base clustering methods may be weak, which may lead to unreliable base clusters, the hypergraph constructed from these unreliable base clusters may also be imperfect. Therefore, the final results, which are obtained by partitioning the unreliable hypergraph, may also be undesirable.

To address this issue, in this paper, we propose a novel Clustering Ensemble method with Structured Hypergraph Learning (CESHL). Different from conventional hypergraph based methods, in our method, the hypergraph is dynamically learned from data instead of directly being pre-defined. As mentioned before, some base clusters may be unreliable which may be harmful to hypergraph learning. In our method,

* Corresponding author at: School of Computer Science and Technology, Anhui University, Hefei 230601, China.
E-mail addresses: zhoupeng@ahu.edu.cn (P. Zhou), e19201043@stu.ahu.edu.cn (X. Wang), duliang@sxu.edu.cn (L. Du), xjli@ahu.edu.cn (X. Li).

https://doi.org/10.1016/j.inffus.2021.09.003
Received 11 March 2021; Received in revised form 2 June 2021; Accepted 15 September 2021
Available online 30 September 2021

we provide a new method to evaluate the quality of the base clusters, and learn the hypergraph by considering the quality of each cluster. Therefore, in the learning process, the hypergraph will become more and more reliable. Moreover, to make the learned hypergraph more appropriate to the clustering task, we impose some constraints to make the hypergraph have a clear clustering structure, i.e, if we want to partition the data into c clusters, the hypergraph will have exact c connective components. Then obtaining the final consensus result is trivial, because we just need to put the instances in the same connective component into a cluster. Therefore, our method is an end-to-end clustering ensemble method, which does not need any uncertain postprocessing, such as kmeans or hypergraph partitioning.

To achieve this, we propose a simple yet effective objective function to dynamically learn the structured graph. Then we apply an iterative algorithm to optimize the introduced objective function. At last, we conduct extensive experiments on the benchmark data sets. When comparing with conventional hypergraph partition based clustering ensemble methods, ours achieves better performance which demonstrates the effectiveness of our structured hypergraph learning strategy. Moreover, when comparing with some state-of-the-art clustering ensemble methods, ours also often outperforms them which shows the superiority of the proposed method.

The remaining parts of this paper are organized as follows. Section 2 reviews some related work. Section 3 introduces our CESHL in detail. Section 4 presents the experimental results. Section 5 concludes this paper.

2. Related work

In this section, we will introduce some related work about clustering ensemble and hypergraph learning. Firstly, we introduce some notations. We use boldface uppercase letter to denote a matrix and boldface lowercase letter to represent a vector. Given a matrix $\mathbf{A}$, we use A_{ij} to denote its (i,j)th element, and use $\mathbf{A}_{i.}$ and $\mathbf{A}_{.i}$ to denote its ith row and ith column, respectively. Given a hypergraph with n instances and k hyperedges, a node v and a hyperedge e in this hypergraph, a matrix $\mathbf{H} \in \mathbb{R}^{n\times k}$, diagonal matrix $\mathbf{W} \in \mathbb{R}^{k\times k}$ and $\mathbf{D} \in \mathbb{R}^{n\times n}$, and a vector $\mathbf{x} \in \mathbf{R}^n$, supposing that v is the pth node and e is the qth hyperedge in the hypergraph, then we use $\mathbf{H}(v,e)$ to denote H_{pq}, $\mathbf{W}(v)$ to denote W_{pp}, $\mathbf{D}(e)$ to denote D_{qq}, $\mathbf{x}(v)$ to denote x_p, respectively, for better readability. Given a vector $\mathbf{v}$, we use $\|\mathbf{v}\|_2$ to denote its ℓ_2 norm, and use $diag(\mathbf{v})$ to denote the diagonal matrix whose diagonal vector is $\mathbf{v}$. Given a matrix $\mathbf{M}$, we use $\|\mathbf{M}\|_F$ to denote its Frobenius norm, and use $diag(\mathbf{M})$ to denote its diagonal vector. $rank(\mathbf{M})$ denotes the rank of matrix $\mathbf{M}$ and $tr(\mathbf{M})$ denotes its trace.

2.1. Clustering ensemble

Given a data set with n instances $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, we can obtain its m base clusterings $\{\mathcal{C}^1, \dots, \mathcal{C}^m\}$, where the jth base clustering $\mathcal{C}^j$ contains k_j base clusters $\pi_1^j, \dots, \pi_{k_j}^j$ and $\mathcal{X} = \bigcup_{i=1}^{k_j} \pi_i^j$. Denoting $k = \sum_{j=1}^m k_j$ as the total number of all base clusters, in the following, we also use $\pi_1, \dots, \pi_k$ to denote all the base clusters of the m base clusterings for simplicity. Clustering ensemble aims to learn a consensus partition $\mathcal{C}$ by ensembling the m base clusterings $\{\mathcal{C}^1, \dots, \mathcal{C}^m\}$ [2,15].

One most related task is multi-view learning [16–20]. Multi-view learning integrates the features in multiple views to obtain a consensus classification or clustering results. For example, Kang et al. constructed multiple graphs from multi-view features and ensembled the multiple graphs for spectral clustering [18]; Tang et al. integrated multiple views by graph diffusion [19]. Different from multi-view learning, which fusions information on the feature level, clustering ensemble often integrates information on the decision level. Clustering ensemble directly fusions the multiple clustering results without accessing the original features of data. Therefore, clustering ensemble is a more challenging task. Moreover, since clustering ensemble does not need the original data, it is helpful to protect the privacy of data [8].

To ensemble multiple base clusterings, there are some representative strategies. One popular strategy is that it reformulates the clustering ensemble problem to a new clustering problem, where each instance is represented by its assignment in the base clustering results instead of its original features. Therefore, it is a clustering problem on the categorical data. For example, Topchy et al. proposed an expectation maximization clustering method for the categorical data [21]; Nguyen et al. applied kmodes method to partition the categorical data [22]; Bai et al. proposed an information theoretical framework for clustering ensemble [23].

Another strategy is to relabel each data by the label alignment methods based on the multiple clustering results. For example, Hore et al. proposed a scalable relabel method for clustering ensemble which is appropriate for big data [24]; Li et al. provided a label alignment method based on Dempster–Shafer evidence theory [25].

Since the clustering results can be represented by graphs or similarity matrix, many methods obtain the final consensus results based on graphs. For example, Fern et al. and Zhou et al. constructed bipartite graphs for clustering ensemble in [3] and [26], respectively; Iam-On et al. proposed a similarity metric based on the link of instances for ensemble [27,28]. Some work constructed a co-association matrix to represent the relationship between instances. For example, Tao et al. applied spectral clustering on the co-association matrix [12,29,30]; Huang et al. proposed a scalable spectral clustering on the co-association matrix [31]; Tao et al. applied adversarial learning on the graph for clustering ensemble [32]. Among these graph based methods, Strehl et al. proposed a hypergraph based ensemble method HGPA (HyperGraph Partitioning Algorithm) [1]. It constructed hypergraph from base clusterings, where each node represented an instance and each hypergraph represented a base cluster. Then it applied HMETIS algorithm [33] to partition the hypergraph to obtain the final clusterings.

Different from this hypergraph partitioning algorithm, which predefines a static hypergraph from base clusterings, our proposed method dynamically learns a structured hypergraph for clustering ensemble. The benefits are two folds. On one hand, due to the low quality of base clusters, the static hypergraph may also be unreliable. However, our dynamically learned hypergraph can become more and more reliable in the process of ensemble learning. On the other hand, the hypergraph we learned has a clearer cluster structure, i.e., it contains exact c connective components, and thus we do not need the postprocess like hypergraph partitioning.

2.2. Hypergraph learning

Given a data set with n instances $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, a hypergraph $\mathcal{G}$ can be represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$. $\mathcal{V}$ is the node set which contains n nodes and each node represents an instance. $\mathcal{E}$ is the hyperedge set which contains k hyperedges, where each hyperedge links multiple instances to represent some relationship among instances. The hyperedge weight matrix $\mathbf{W} \in \mathbb{R}^{k\times k}$ is a diagonal matrix where each diagonal element represents the weight of each hyperedge.

Then, we can construct its incidence matrix $\mathbf{H} \in \mathbb{R}^{n\times k}$. For any node $v \in \mathcal{V}$ and any hyperedge $e \in \mathcal{E}$, $\mathbf{H}(v,e)$ shows the connection between the node v and hyperedge e. The node degree matrix $\mathbf{D}_v \in \mathbb{R}^{n\times n}$ is a diagonal matrix whose diagonal element is the degree of each instance, and it can be formally defined as $\mathbf{D}_v = diag(\mathbf{HW1})$, where $\mathbf{1}$ is a vector whose elements are all 1's. The hyperedge degree matrix $\mathbf{D}_e \in \mathbb{R}^{k\times k}$ is also a diagonal matrix and its diagonal element is the degree of each hyperedges. $\mathbf{D}_e$ can be formally defined as $\mathbf{D}_e = diag(\mathbf{1}^T\mathbf{H})$. Then we can define its Laplacian matrix as $\mathbf{L} = \mathbf{I} - \mathbf{D}_v^{-\frac{1}{2}}\mathbf{HWD}_e^{-1}\mathbf{H}^T\mathbf{D}_v^{-\frac{1}{2}}$, where $\mathbf{I}$ is the identity matrix.

Since hypergraph can characterize the high-order relationship among instances, it has attracted much attention in recent years [14,

Table 1
Main notations and descriptions used in CESHL.

Notation	Description
n	Number of instances.
k	Number of base clusters.
$\mathbf{H} \in \mathbb{R}^{n\times k}$	The incidence matrix of the initial hypergraph.
$\mathbf{W} \in \mathbb{R}^{k\times k}$	The diagonal matrix whose diagonal elements are the weights of the hyperedges.
$\mathbf{Y} \in \mathbb{R}^{n\times k}$	The incidence matrix of the structured hypergraph.
$\mathbf{D}_v \in \mathbb{R}^{n\times n}$	The node degree matrix of the structured hypergraph.
$\mathbf{D}_e \in \mathbb{R}^{k\times k}$	The hyperedge degree matrix of the structured hypergraph.
$\mathbf{L} \in \mathbb{R}^{n\times n}$	The Laplacian matrix of the structured hypergraph.

34–37]. Among them, many works propagate information on hypergraphs to fulfill some machine learning and pattern recognition tasks. For example, Yu et al. used hypergraph to characterize the high-order information among instances and applied it to the image classification task [16]; Purkait et al. applied hypergraph to clustering on large scale data [38]; Zhao et al. used hypergraph for social network embedding [37].

The above-mentioned methods just apply hypergraph to some tasks, and do not learn the structure of hypergraph. Different from these works, few works pay attention to how to learn such hypergraph. For example, Gao et al. learned the hyperedge weight matrix **W** and applied it to 3-D object retrieval [39]; Zhang et al. learned the incidence matrix **H** for a semi-supervised embedding task [36]; Yu et al. applied hypergraph learning to the supervised scenario such as image classification [40]; Yu et al. further extended the hypergraph learning to multi-view scenario to handle image re-ranking task [41]; Tang et al. learned an adaptive hypergraph for semi-supervised multi-label image annotation [42].

Our proposed work also dynamically learns the incidence matrix **H**. Different from the previous works, we learn a structured hypergraph, whereas previous works do not impose any constraints on the structure of hypergraphs. Moreover, the goal of the previous works is to learn some embedding of the hypergraph. However, our method just aims to learn the hypergraph itself, since the final clustering results can be trivially obtained from the structured hypergraph.

3. Clustering ensemble via structured hypergraph learning

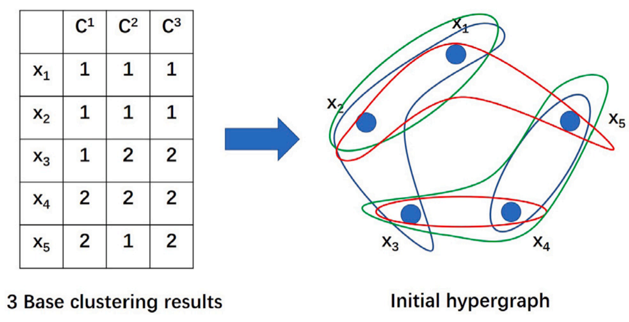
In this section, we will introduce our CESHL in more detail. We first introduce some main notations of CESHL in Table 1.

3.1. Constructing the initial hypergraph

Given multiple base clustering results $C^1, \dots, C^m$ of n instances, where base clustering result C^j contains k_j clusters $(\pi_1^j, \dots, \pi_{k_j}^j)$, we first construct an initial hypergraph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{W}\}$ from them. In more detail, in the node set $\mathcal{V}$, each node represents an instance, and thus there are n nodes totally. Each hyperedge in $\mathcal{E}$ represents a base cluster π_i^j. Therefore, there are $k = \sum_{j=1}^{m} k_j$ hyperedges in total. The edge weight matrix $\mathbf{W} \in \mathbb{R}^{k\times k}$ is a diagonal matrix whose diagonal element W_{ii} denotes the weight of the ith hyperedge. We will introduce how to construct **W** later.

Fig. 1 shows a simple example. In this example, there are 5 instances $(\mathbf{x}_1, \dots, \mathbf{x}_5)$ and 3 base clustering results (C^1, C^2, C^3). The left side of Fig. 1 shows the base clustering results. For example, in the first base clustering C^1, $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ belong to the first cluster, and $\mathbf{x}_4$ and $\mathbf{x}_5$ belong to the second cluster. The right side of Fig. 1 shows the corresponding hypergraph, which has 5 nodes (blue circles) and 6 hyperedges (closed curves with different colors). The two blue closed curves denote the two hyperedges generated from C^1, the red ones denote the two hyperedges generated from C^2, and the greens represent the two hyperedges generated from C^3.

Then we can construct the incidence matrix $\mathbf{H} \in \mathbb{R}^{n\times k}$ from $\mathcal{G}$. Since it is just an initial hypergraph, and we will relearn it in the following steps, we can initialize **H** very easily as $\mathbf{H}(v, e) = 1$ if $v \in e$ and



	C¹	C²	C³
x₁	1	1	1
x₂	1	1	1
x₃	1	2	2
x₄	2	2	2
x₅	2	1	2

Fig. 1. An illustration of constructing initial hypergraph from base clusterings.

$\mathbf{H}(v, e) = 0$ otherwise. Taking Fig. 1 as an example, the incidence matrix **H** of the hypergraph in Fig. 1 is

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

.

3.2. Evaluating the edge weight matrix W

Now we will focus on how to construct the edge weight matrix **W**. As introduced before, **W** is a diagonal matrix. For simplicity, we denote the diagonal vector of **W** as $\mathbf{w} \in \mathbb{R}^k$, where w_i denotes the weight of the ith hyperedge.

Since each hyperedge represents a cluster in base clusterings, to evaluate the weight of the hyperedge is equivalent to evaluate the quality of each base cluster. Therefore, we focus on how to evaluate the quality of each base cluster $(\pi_1, \dots, \pi_k)$. Since we do not access the original data, we can only evaluate the quality of clusters via the consistency of the base clusterings. In more details, if one base cluster is consistent with many other base clusters, then we believe its quality is high. Therefore, given a base cluster, we check the appearance of all pairs $(\mathbf{x}_p, \mathbf{x}_q)$ in the cluster. The more times the pairs appear in other base clusters, the more consistent the base cluster is with others, and thus the higher the quality of this cluster is. More formally, we define the co-association matrix **U** as $\mathbf{U} = \frac{1}{m}\mathbf{H}\mathbf{H}^T$. Then, for any pair $(\mathbf{x}_p, \mathbf{x}_q)$, the larger U_{pq} is, the more times it appears in base clusters. For any base cluster π_i, we can compute its initial quality score, denoted by s_i, by computing the mean of all U_{pq} in π_i, i.e.,

$$s_i = \frac{1}{n_i^2} \sum_{\mathbf{x}_p \in \pi_i, \mathbf{x}_q \in \pi_i} U_{pq}, \tag{1}$$

where n_i is the number of instances in the cluster π_i.

Of course, we can use $\mathbf{s} = [s_1, \dots, s_k]^T$ as the quality vector, i.e., $\mathbf{w} = \mathbf{s}$. However, this score considers each base cluster independently and ignores the relativity between two clusters. For example, if π_i has a very high quality and π_j is very similar to π_i, then π_j should also have a high quality. To this end, we need a simple cluster-wise similarity matrix $\mathbf{C} \in \mathbb{R}^{k\times k}$ defined as $\mathbf{C} = \mathbf{H}^T\mathbf{H}$.

Inspired by the famous manifold ranking [43], we have that, on one hand, $\mathbf{w}$ should be consistent with $\mathbf{s}$, and thus we need to minimize $\|\mathbf{w}-\mathbf{s}\|_2^2$; on the other hand, if π_i is similar to π_j, (i.e., C_{ij} is large), then w_i should be close to w_j, and thus we can minimize $\frac{C_{ij}}{2}\left(\frac{w_i}{\sqrt{\sum_{p=1}^k C_{ip}}} - \frac{w_j}{\sqrt{\sum_{p=1}^k C_{jp}}}\right)^2$ to achieve this. The denominators are the normalization to eliminate the differences caused by the different scales of $\mathbf{C}$ as suggested in [43]. Thus the objective function of learning $\mathbf{w}$ is:

$$\min_{\mathbf{w}} \sum_{i,j=1}^{k} \frac{C_{ij}}{2}\left(\frac{w_i}{\sqrt{\sum_{p=1}^k C_{ip}}} - \frac{w_j}{\sqrt{\sum_{p=1}^k C_{jp}}}\right)^2 + \lambda\|\mathbf{w}-\mathbf{s}\|_2^2, \qquad s.t.\quad \forall i: 0 \le w_i \le 1. \tag{2}$$

where λ is a balanced hyper-parameter. The constraint is to make sure that the weight is in the range $[0,1]$. To optimize Eq. (2), we can reformulate the first term as:

$$\begin{aligned}\sum_{i,j=1}^{k} \frac{C_{ij}}{2}\left(\frac{w_i}{\sqrt{\sum_{p=1}^k C_{ip}}} - \frac{w_j}{\sqrt{\sum_{p=1}^k C_{jp}}}\right)^2 &= \sum_{i,j=1}^{k} \frac{C_{ij} w_i^2}{\sum_{p=1}^k C_{ip}} \\ &\quad - \sum_{i,j=1}^{k} \frac{C_{ij} w_i w_j}{\sqrt{\sum_{p=1}^k C_{ip} \sum_{p-1}^k C_{jp}}} \\ &= \sum_{i=1}^{k} w_i^2 - \sum_{i,j=1}^{k} w_i \frac{1}{\sqrt{\sum_{p=1}^k C_{ip}}} \\ &\quad \times C_{ij} \frac{1}{\sqrt{\sum_{p=1}^k C_{jp}}} w_j \\ &= \mathbf{w}^T(\mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{C}\mathbf{D}^{-\frac{1}{2}})\mathbf{w} \end{aligned} \tag{3}$$

where $\mathbf{D}$ is a diagonal matrix whose diagonal element $D_{ii} = \sum_{p=1}^k C_{ip}$. Then Eq. (2) can be reformulate as:

$$\min_{\mathbf{w}} \mathbf{w}^T(\mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{C}\mathbf{D}^{-\frac{1}{2}})\mathbf{w} + \lambda\|\mathbf{w}-\mathbf{s}\|_2^2, \qquad s.t.\quad \forall i: 0 \le w_i \le 1. \tag{4}$$

It is easy to verify that Eq. (4) is a convex quadratic programming problem and can be optimized by some standard methods, such as trust region reflective algorithm. In our implication, we use *quadprog* function provided in Matlab. The time complexity of solving this problem is $O(k^3)$. After obtaining $\mathbf{w}$ by optimizing Eq. (4), we construct $\mathbf{W} = diag(\mathbf{w})$.

3.3. Learning the structured hypergraph

After obtaining $\mathbf{H}$ and $\mathbf{W}$, we obtain the initial hypergraph $\mathcal{G}$ completely. Then, we aim to learn the final consensus clustering result from the hypergraph. Ideally, if we want to partition the data into c clusters, we wish the hypergraph contains exact c connective components, and we just need to put all instances in one connective component into a cluster. However, in real applications, due to the low quality of base clustering results, the numbers of connective components of the initial hypergraph will not be exact c. Take Fig. 1 as an example again, assume that we wish to partition the data into 2 clusters, but we find that the hypergraph only contains one connective component, i.e., all instances are entangled together because some base clusters may contain noises. To learn the structured hypergraph, we need to adjust the incidence matrix $\mathbf{H}$ to reveal the clustering structure. Fig. 2 is an illustration of the structured hypergraph learning. In Fig. 2, the left side is the initial hypergraph constructed in Fig. 1, and the right side is the structured hypergraph $\mathcal{G}'$. We find that we just need to adjust 2 hyperedges

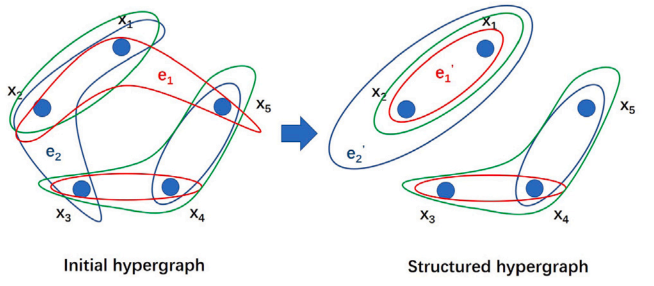


Fig. 2. An illustration of constructing structured hypergraph.

(i.e., e_1 and e_2 in $\mathcal{G}$), and we can obtain structured hypergraph $\mathcal{G}'$ which contains 2 connective components.

Denote $\mathbf{Y} \in [0,1]^{n\times k}$ as the incidence matrix of the structured hypergraph $\mathcal{G}'$. We need to learn $\mathbf{Y}$ from the initial incidence matrix $\mathbf{H}$ and edge weight matrix $\mathbf{W}$. For the structured hypergraph, we can compute its node degree matrix $\mathbf{D}_v = diag(\mathbf{YW1})$ and edge degree matrix $\mathbf{D}_e = diag(\mathbf{1}^T\mathbf{Y})$. Then, we define the Laplacian matrix

$$\mathbf{L} = \mathbf{I} - \mathbf{D}_v^{-\frac{1}{2}}\mathbf{Y}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{Y}^T\mathbf{D}_v^{-\frac{1}{2}}. \tag{5}$$

The following Theorem shows the relation between the rank of Laplacian matrix and the number of connective components.

Theorem 1. *Given any hypergraph $\mathcal{G} = \{\mathcal{V},\mathcal{E},\mathbf{W}\}$ with n nodes, if the rank of its Laplacian matrix $\mathbf{L}$, which is defined as Eq. (5), is $n-c$, then $\mathcal{G}$ contains exact c connective components.*

Proof. See Appendix A. □

According to Theorem 1, to make $\mathcal{G}'$ contain c connective components, we just need a constraint that $rank(\mathbf{L}) = n-c$. Therefore, we can learn its incidence matrix $\mathbf{Y}$ by the following objective function:

$$\min_{\mathbf{Y}} \|\mathbf{Y}-\mathbf{H}\|_F^2, \qquad s.t.\quad rank(\mathbf{L}) = n-c, \quad \forall i,j: 0 \le Y_{ij} \le 1. \tag{6}$$

The loss function is to make $\mathbf{Y}$ fit the initial incidence matrix $\mathbf{H}$. However, this loss function treats all hyperedges (or equivalently speaking, base clusters) the same, which may be inappropriate, because some unreliable base clusters may mislead the hypergraph learning. Intuitively, some hyperedges in $\mathbf{H}$ are good and $\mathbf{Y}$ should prone to fit them; and some hyperedges have low quality and thus $\mathbf{Y}$ does not need to fit them. Therefore, we can use $\mathbf{W}$ learned by previous subsection to weight the loss function Eq. (6), and obtain our final formula:

$$\min_{\mathbf{Y}} \|(\mathbf{Y}-\mathbf{H})\mathbf{W}\|_F^2, \qquad s.t.\quad rank(\mathbf{L}) = n-c, \quad \forall i,j: 0 \le Y_{ij} \le 1. \tag{7}$$

Note that $\mathbf{W}$ is a diagonal matrix, and thus Eq. (7) is equivalent to impose weight w_i on the ith column of $\mathbf{Y}-\mathbf{H}$. Large w_i, which means the ith hyperedge has a high quality, will force that $\mathbf{Y}_{.i}$ should be close to $\mathbf{H}_{.i}$.

3.4. Optimization

Now we introduce how to optimize objective function Eq. (7). Firstly, we need to handle the rank constraint. Since the rank of $\mathbf{L}$ is $n-c$, the c smallest eigenvalues of $\mathbf{L}$ should be zeros. To achieve this, we minimize $\sum_{i=1}^c \sigma_i(\mathbf{L})$, where $\sigma_i(\mathbf{L})$ denotes the ith smallest eigenvalues of $\mathbf{L}$. According to Ky Fan Theorem [44], by introducing an orthogonal auxiliary matrix $\mathbf{F} \in \mathbb{R}^{n\times c}$, we have $\sum_{i=1}^c \sigma_i(\mathbf{L}) = \min_{\mathbf{F}^T\mathbf{F}=\mathbf{I}} tr(\mathbf{F}^T\mathbf{L}\mathbf{F})$. Therefore, Eq. (7) can be rewritten as the following form:

$$\min_{\mathbf{Y},\mathbf{F}} \|(\mathbf{Y}-\mathbf{H})\mathbf{W}\|_F^2 + \rho\, tr(\mathbf{F}^T\mathbf{L}\mathbf{F}), \tag{8}$$

$$s.t.\quad \mathbf{F}^T\mathbf{F}=\mathbf{I},\quad \forall i,j: 0\le Y_{ij}\le 1,$$

where ρ is a large enough parameter to control the rank of $\mathbf{L}$. Then we will optimize Eq. (8) instead. Since Eq. (8) contains two variables $\mathbf{Y}$ and $\mathbf{F}$, we optimize one of them respectively while fixing the other one.

3.4.1. *Optimize* $\mathbf{F}$

When optimizing $\mathbf{F}$, Eq. (8) can be simplified as

$$\min_{\mathbf{F}} \quad tr(\mathbf{F}^T\mathbf{L}\mathbf{F}), \qquad s.t.\quad \mathbf{F}^T\mathbf{F}=\mathbf{I}. \tag{9}$$

It can be optimized by Ky Fan Theorem. $\mathbf{F}$ contains the c eigenvectors corresponding to the smallest c eigenvalues of $\mathbf{L}$.

3.4.2. *Optimize* $\mathbf{Y}$

When optimizing $\mathbf{Y}$, we need to take Eq. (5) into Eq. (8), and obtain the following subproblem:

$$\min_{\mathbf{Y}} \quad \mathcal{J}=\|(\mathbf{Y}-\mathbf{H})\mathbf{W}\|_F^2-\rho tr(\mathbf{F}^T\mathbf{D}_v^{-\frac{1}{2}}\mathbf{Y}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{Y}^T\mathbf{D}_v^{-\frac{1}{2}}\mathbf{F}), \qquad s.t.\quad \forall i,j: 0\le Y_{ij}\le 1. \tag{10}$$

Note that $\mathbf{D}_v$ and $\mathbf{D}_e$ depend on $\mathbf{Y}$, and thus Eq. (10) is complex. We apply Quasi-Newton method to optimize it. We first need to compute the partial derivative of $\mathcal{J}$ w.r.t. $\mathbf{Y}$. Denote that $\mathbf{D}_v^{-\frac{1}{2}}=diag([d_1^v,\dots,d_n^v])$ and $\mathbf{D}_e^{-1}=diag([d_1^e,\dots,d_n^e])$, we have

$$\frac{\partial d_i^v}{\partial Y_{pq}}=\frac{\partial\left(\sum_{j=1}^k Y_{ij}w_j\right)^{-\frac{1}{2}}}{\partial Y_{pq}}=-\frac{1}{2}(d_i^v)^3\delta_{ip}w_q, \tag{11}$$

and

$$\frac{\partial d_j^e}{\partial Y_{pq}}=\frac{\partial\left(\sum_{i=1}^n Y_{ij}\right)^{-1}}{\partial Y_{pq}}=-(d_j^e)^2\delta_{jq}, \tag{12}$$

where δ is the Kronecker delta, i.e., $\delta_{ip}=1$ if $i=p$ and $\delta_{ip}=0$ otherwise. Taking Eqs. (11) and (12) into the partial derivative of $\mathcal{J}$ w.r.t. $\mathbf{Y}$, and according to the chain rule, we have

$$\begin{aligned}\frac{\partial\mathcal{J}}{\partial\mathbf{Y}}=&2(\mathbf{Y}-\mathbf{H})\mathbf{W}^2+\rho\mathbf{J}diag\left(\mathbf{Y}^T\mathbf{D}_v^{-\frac{1}{2}}\mathbf{F}\mathbf{F}^T\mathbf{D}_v^{-\frac{1}{2}}\mathbf{Y}\mathbf{W}\mathbf{D}_e^{-2}\right)\\&+\rho\mathbf{D}_v^{-\frac{3}{2}}diag\left(\mathbf{Y}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{Y}^T\mathbf{D}_v^{-\frac{1}{2}}\mathbf{F}\mathbf{F}^T\right)\mathbf{J}\mathbf{W}-2\rho\mathbf{D}_v^{-\frac{1}{2}}\mathbf{F}\mathbf{F}^T\mathbf{D}_v^{-\frac{1}{2}}\mathbf{Y}\mathbf{W}\mathbf{D}_e^{-1}\end{aligned} \tag{13}$$

where $\mathbf{J}$ is a matrix whose elements are all 1's. Although Eq. (13) seems complex, since $\mathbf{D}_v$, $\mathbf{D}_e$ and $\mathbf{W}$ are diagonal matrices, it can be computed in $O(nkc)$ time by appropriately using the associative law of matrix multiplication. At the end of each iteration in the Quasi-Newton method, if Y_{ij} is not in the range $[0,1]$, we project it into the range $[0,1]$ by $Y_{ij}=\min(\max(Y_{ij},0),1)$.

Note that there is a hyper-parameter ρ which controls the rank of $\mathbf{L}$. We initialize $\rho=100$ and adjust ρ by observing the rank of $\mathbf{L}$. If $rank(L)>n-c$, which means the constraint is a little too weak, we increase $\rho\leftarrow 2\rho$; and if $rank(L)<n-c$, which means the constraint is too strong, we decrease it by $\rho\leftarrow\rho/2$.

Algorithm 1 summarizes the whole process of our CESHL.

3.5. *Complexity analysis*

Constructing the co-association matrix $\mathbf{U}$ costs $O(n^2m)$ time and computing $\mathbf{s}$ costs $O(n^2k/c)$ time. It takes $O(k^2n)$ time to compute the cluster-wise similarity matrix $\mathbf{C}$. Then, solving the convex quadratic programming problem Eq. (4) takes $O(k^3)$ time as introduced before. When iteratively optimizing Eq. (8), we use L-BFGS algorithm [45] (one of the most popular kinds of Quasi-Newton method) whose time complexity in each iteration is $O(nr)$, where r is the number of steps stored in memory which can be viewed as a constant. Computing the gradient in each iteration costs $O(nkc)$ time as introduced before. Therefore, the time complexity of solving Eq. (10) is $O(t_1(nr+nkc))$, where t_1 is the number of iterations in L-BFGS algorithm. Solving Eq. (9) by eigenvalue decomposition costs $O(n^2c)$ time. Assuming the number of iterations of lines 6–9 in Algorithm 1 is t_2, the whole time complexity of Algorithm 1 is $O(n^2m+k^2n+k^3+t_2(t_1nkc+n^2c))$. Note that, in the real applications, we often have $n\gg k$, and thus the time complexity is often quadratic in n, which is comparable with many other clustering ensemble methods such as [30,46–48]. In addition, our algorithm converges very quickly (often within 20 iterations).

Algorithm 1: CESHL Algorithm.

```
Input: $m$ base clustering results $C_1,\cdots,C_m$, the number $c$ of clusters, and
    hyper-parameter $\lambda$.
Output: Final consensus clustering result.
1:  Construct initial incidence matrix $\mathbf{H}$.
2:  Compute the initial quality scores $\mathbf{s}$ of base clusters by Eq. (1).
3:  Compute the cluster-wise similarity matrix $\mathbf{C}$ by $\mathbf{C}=\mathbf{H}^T\mathbf{H}$.
4:  Compute the weight matrix of hyperedges $\mathbf{W}$ by solving Eq. (4).
5:  //Iteratively optimizing Eq. (8).
6:  while not converge do
7:      Optimize $\mathbf{F}$ by solving Eq. (9) with eigenvalue decomposition of
        $\mathbf{L}$.
8:      Optimize $\mathbf{Y}$ by solving Eq. (10) with Quasi-Newton method.
9:  end while
10: Obtain the final clusters from the $c$ connective component in $\mathbf{Y}$.
```

Table 2
Description of the data sets.

	# of instances	# of features	# of classes
Arcene	200	10000	2
Glioma	50	4434	4
K1b	2340	21839	6
Lung	203	3312	5
MNIST4000	4000	784	10
ORL	400	1024	40
Orlraws	100	10304	10
Tr41	878	7454	10

4. Experiments

In this section, we compare our CESHL with some state-of-the-art clustering ensemble methods on benchmark data sets.

4.1. *Data sets*

We use 8 data sets, including Arcene,[1] Glioma,[1] K1b [49], Lung,[1] MNIST4000,[2] ORL,[1] Orlraws,[1] Tr41 [49], whose detailed information is shown in Table 2.

4.2. *Experimental setup*

Following the similar experimental setting in [46], we run kmeans on each data set 200 times to obtain 200 base clustering results. Then we divide them into 10 subsets, where each subset contains 20 base results. Then we run the ensemble methods on each subset (i.e., we ensemble 20 base results each time) and obtain 10 ensemble results. We report the average result of the 10 ensemble results. We compare our CESHL with the following methods:

- **KM(avg.)**. It is the average result of the base kmeans results.

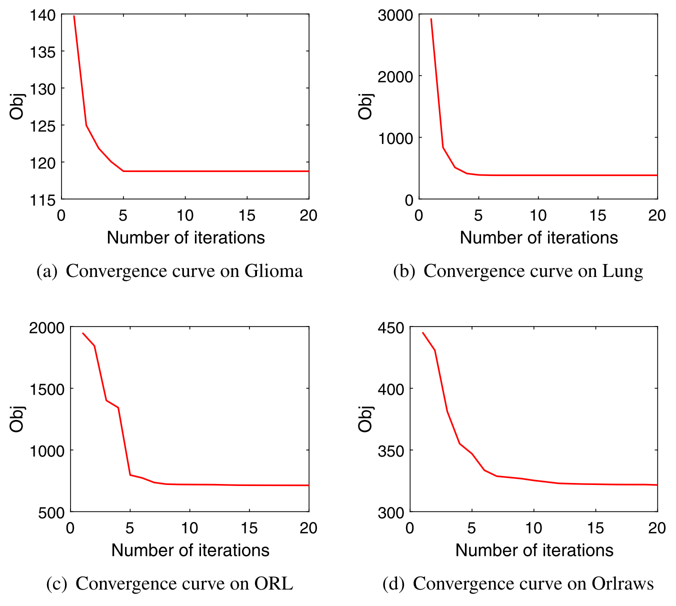
[1] https://jundongl.github.io/scikit-feature/datasets.html
[2] http://www.cad.zju.edu.cn/home/dengcai/Data/data.html

Table 3
ACC results on all the data sets.

Methods	Arcene	GLIOMA	K1b	Lung	MNIST4000	ORL	Orlraws	Tr41
KM(avg.)	0.6215 ±0.0123	0.4260 ±0.0092	0.6551 ±0.0334	0.5430 ±0.0162	0.5101 ±0.0065	0.5034 ±0.0103	0.6485 ±0.0452	0.5424 ±0.0286
KM(best)	0.6500 ±0.0000	0.4900 ±0.0302	0.7944 ±0.0370	0.6453 ±0.0310	0.5544 ±0.0128	0.5583 ±0.0124	0.7810 ±0.0509	0.6630 ±0.0415
CSPA [1]	0.6395 ±0.0076	0.4160 ±0.0280	0.4550 ±0.0052	0.4837 ±0.0031	0.5226 ±0.0264	0.5727 ±0.0292	0.7660 ±0.0534	0.5059 ±0.0249
HGPA [1]	0.5100 ±0.0000	0.4400 ±0.0462	0.4984 ±0.0810	0.4690 ±0.0500	0.1027 ±0.0000	0.6010 ±0.0245	0.7950 ±0.0467	0.4926 ±0.0530
MCLA [1]	0.6410 ±0.0248	0.4260 ±0.0190	0.6777 ±0.0843	0.5438 ±0.0153	0.5127 ±0.0181	0.5935 ±0.0288	0.7930 ±0.0302	0.5677 ±0.0451
NMFC [50]	0.6260 ±0.0310	0.4240 ±0.0227	0.6453 ±0.0566	0.5581 ±0.0427	0.5152 ±0.0153	0.5818 ±0.0267	0.7690 ±0.0559	0.5961 ±0.0416
RCE [46]	0.6440 ±0.0190	0.4100 ±0.0216	0.6810 ±0.0604	0.5325 ±0.0230	0.5272 ±0.0081	0.6075 ±0.0146	0.7974 ±0.0282	0.6216 ±0.0197
LWEA [51]	0.6080 ±0.0290	0.4200 ±0.0231	0.7977 ±0.0653	0.5084 ±0.0387	0.5174 ±0.0359	0.5670 ±0.0195	0.7570 ±0.0531	0.6562 ±0.0375
LWGP [51]	0.6140 ±0.0310	0.4380 ±0.0290	0.7109 ±0.0782	0.5281 ±0.0148	0.5197 ±0.0157	0.5935 ±0.0106	0.7750 ±0.0272	0.6379 ±0.0238
RSEC [30]	0.6045 ±0.0332	0.4260 ±0.0443	0.7468 ±0.0761	0.5719 ±0.0664	0.2612 ±0.0659	0.4508 ±0.0231	0.6110 ±0.0674	0.4141 ±0.0497
DREC [47]	0.6320 ±0.0290	0.4220 ±0.0239	0.6063 ±0.0665	0.5271 ±0.0374	0.5297 ±0.0246	0.5973 ±0.0124	0.7880 ±0.0282	0.6246 ±0.0416
SPCE [48]	0.6440 ±0.0359	0.4160 ±0.0519	0.8120 ±0.0337	0.7340 ±0.0630	0.1822 ±0.0621	0.5558 ±0.0529	0.7750 ±0.1084	0.6603 ±0.1485
SCCBG [26]	0.5470 ±0.0747	0.4160 ±0.1770	0.8429 ±0.0921	0.6483 ±0.1001	0.4591 ±0.0754	0.5627 ±0.0352	0.7690 ±0.0681	0.6129 ±0.0687
CESHL	**0.6450** ±0.0196	**0.4660** ±0.0267	**0.8544** ±0.0505	**0.8020** ±0.0252	**0.5341** ±0.0088	**0.6120** ±0.0121	**0.8250** ±0.0299	**0.6845** ±0.0511

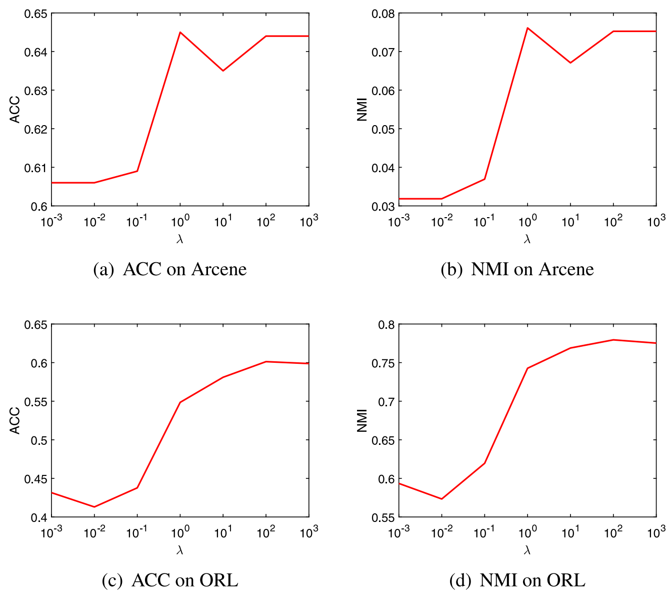
Table 4
NMI results on all the data sets.

Methods	Arcene	GLIOMA	K1b	Lung	MNIST4000	ORL	Orlraws	Tr41
KM(avg.)	0.0512 ±0.0118	0.1643 ±0.0072	0.5358 ±0.0210	0.3910 ±0.0144	0.4741 ±0.0026	0.7148 ±0.0082	0.7382 ±0.0308	0.5667 ±0.0253
KM(best)	0.0818 ±0.0281	0.2456 ±0.0000	0.6322 ±0.0312	0.4458 ±0.0306	0.5061 ±0.0088	0.7468 ±0.0072	0.8194 ±0.0356	0.6490 ±0.0361
CSPA [1]	0.0584 ±0.0066	0.1804 ±0.0470	0.4045 ±0.0066	0.3336 ±0.0064	0.4461 ±0.0125	0.7609 ±0.0118	0.8082 ±0.0457	0.5856 ±0.0201
HGPA [1]	0.0003 ±0.0000	0.1647 ±0.0424	0.3285 ±0.1013	0.3060 ±0.0488	0.0000 ±0.0000	0.7788 ±0.0093	0.8297 ±0.0291	0.4731 ±0.0452
MCLA [1]	0.0718 ±0.0248	0.1747 ±0.0310	0.5643 ±0.0536	0.3866 ±0.0152	0.4619 ±0.0117	0.7683 ±0.0179	0.8295 ±0.0207	0.6001 ±0.0225
NMFC [50]	0.0557 ±0.0337	0.1681 ±0.0267	0.5213 ±0.0244	0.4024 ±0.0223	0.4695 ±0.0251	0.7675 ±0.0099	0.8253 ±0.0235	0.6329 ±0.0309
RCE [46]	0.0752 ±0.0206	0.1650 ±0.0344	0.5792 ±0.0391	0.3902 ±0.0216	0.4863 ±0.0094	0.7798 ±0.0044	0.8445 ±0.0200	0.6441 ±0.0209
LWEA [51]	0.0361 ±0.0315	0.1466 ±0.0140	0.6578 ±0.0610	0.3482 ±0.0242	0.4463 ±0.1032	0.7707 ±0.0084	0.8221 ±0.0203	0.6524 ±0.0193
LWGP [51]	0.0426 ±0.0337	0.1774 ±0.0268	0.6009 ±0.0442	0.3774 ±0.0164	0.4523 ±0.0838	0.7792 ±0.0074	0.8230 ±0.0229	0.6577 ±0.0248
RSEC [30]	0.0349 ±0.0325	0.1513 ±0.0345	0.4909 ±0.0986	0.3778 ±0.0643	0.1907 ±0.0774	0.6605 ±0.0142	0.6736 ±0.0502	0.3418 ±0.0741
DREC [47]	0.0622 ±0.0315	0.1740 ±0.0354	0.5512 ±0.0530	0.3709 ±0.0114	0.4756 ±0.0189	0.7764 ±0.0069	0.8409 ±0.0179	0.6505 ±0.0306
SPCE [48]	0.0752 ±0.0317	**0.2649** ±0.0189	0.6434 ±0.0833	0.4489 ±0.1736	0.0968 ±0.0850	0.7863 ±0.0691	0.8336 ±0.0779	0.6292 ±0.2116
SCCBG [26]	**0.0761** ±0.0211	0.1770 ±0.0435	0.6384 ±0.2086	0.4463 ±0.0714	0.4379 ±0.0790	0.7415 ±0.0243	0.8131 ±0.0479	0.6280 ±0.0421
CESHL	**0.0761** ±0.0211	0.1938 ±0.0317	**0.6793** ±0.1271	**0.5458** ±0.0363	**0.4905** ±0.0046	**0.7880** ±0.019	**0.8476** ±0.0200	**0.6630** ±0.0433



(a) Convergence curve on Glioma (b) Convergence curve on Lung (c) Convergence curve on ORL (d) Convergence curve on Orlraws

Fig. 3. Convergence curves of CESHL on Glioma, Lung, ORL, and Orlraws.



(a) ACC on Arcene (b) NMI on Arcene (c) ACC on ORL (d) NMI on ORL

Fig. 4. ACC and NMI with respect to λ on Arcene and ORL.

- **KM(best)**. It is the best result among the 20 base kmeans results.
- **CSPA** [1]. It constructs a pairwise similarity based on the relation between the instances in the same cluster.
- **HGPA** [1]. It is a hypergraph partitioning method for clustering ensemble.
- **MCLA** [1]. It transforms the clustering ensemble problem into a cluster correspondence problem.
- **NMFC** [50]. It is a nonnegative matrix factorization based ensemble method.
- **RCE** [46]. It is a robust clustering ensemble method with connective matrices.
- **LWEA** [51]. It is a hierarchical agglomerative clustering ensemble method with locally weighting.
- **LWGP** [51]. It is a graph partitioning based clustering ensemble method with locally weighting.
- **RSEC** [30]. It is a robust spectral clustering ensemble method.
- **DREC** [47]. It is a clustering ensemble method based on dense representation.
- **SPCE** [48]. It is a self-paced clustering ensemble method.
- **SCCBG** [26]. It is a clustering ensemble method with bipartite graph learning.

We use Accuracy (ACC) and Normalized Mutual Information (NMI), which are widely used in clustering tasks, to evaluate the ensemble results. For all methods and all data sets, we set the number of clusters c as the true number of classes. In our method, we tune the hyper-parameter λ in the range $\{10^{-3}, 10^{-2}, \dots, 10^{3}\}$.

4.3. Experimental results

Tables 3 and 4 show the ACC and NMI results of our method and other compared ensemble methods on all data sets. The tables show the average results and the standard deviation over the 10 subsets. The best results of ensemble methods (i.e., except KM(avg.) and KM(best)) are in boldface. From Tables 3 and 4, we find some interesting points:

- Compared with HGPA, which is a hypergraph partitioning based clustering ensemble method, our CESHL outperforms it significantly on all data sets. HGPA constructs a static hypergraph from multiple based results directly, and thus unreliable base results may lead to the poor quality of the hypergraph. Different from HGPA, our method dynamically learns a structured hypergraph, in which way the hypergraph may be more appropriate for the clustering task. That is why our method can easily outperform this static hypergraph partitioning method.
- Besides HGPA, our CESHL also performs better than other state-of-the-art clustering ensemble methods on most data sets. It well demonstrates the effectiveness and superiority of our structured hypergraph learning strategy. Especially compared with the graph based or co-association matrix based methods, such as RCE, LWGP, RSEC, SPCE, our hypergraph based method often achieves better results. The reason may be that the hypergraph can better characterize the complex high-order relation between data compared with the normal graph.
- CESHL outperforms KM(avg.) on all data sets, which shows the benefit of clustering ensemble. Many ensemble methods cannot outperform KM(best). It may be because that among the base clustering results, there are many unreliable base results, which may mislead the ensemble learning. However, our CESHL is usually closed to or even better than KM(best) on most data sets. The reason may be that, in our method we evaluate the quality of each base clusters and in the dynamical hypergraph learning process, we lower the weights of the base clusters with poor quality. The side effect of the unreliable results may be reduced in our method. Note that, CESHL does not need to perform an exhaustive search on the predefined pool of base clusterings, which shows its superiority.

Fig. 3 shows the convergence curves of CESHL on Glioma, Lung, ORL, and Orlraws. The results on other data sets are similar. From Fig. 3, we can see that CESHL often converges within 20 iterations, which demonstrates the claim in Section 3.5.

4.4. Ablation study

To show the effectiveness of the strategy of evaluating the quality of base clusters, we compare CESHL with the following two degenerated versions:

- **CESHL-W**. It drops the edge weight matrix $\mathbf{W}$ (or equivalently speaking, it sets $\mathbf{W} = \mathbf{I}$).
- **CESHL-s**. It sets $\mathbf{W} = diag(\mathbf{s})$, i.e., we use the initial quality score $\mathbf{s}$ as the weights of base clusters without the propagation process.

Table 5
Clustering results compared with degenerated versions.

Methods	Metric	Arcene	GLIOMA	K1b	Lung	MNIST4000	ORL	Orlraws	Tr41
CESHL-W	ACC	0.6390 ±0348	0.3900 ±0.0368	0.8070 ±0.1066	0.6158 ±0.1140	0.5272 ±0.0115	0.5355 ±0.0515	0.7440 ±0.0975	0.5927 ±0.0758
	NMI	0.0753 ±0.0204	0.1247 ±0.0527	0.5706 ±0.2534	0.4404 ±0.0656	0.4845 ±0.0102	0.7131 ±0.0504	0.7915 ±0.0747	0.5512 ±0.1405
CESHL-s	ACC	0.6440 ±0.0190	0.4260 ±0.0378	0.7960 ±0.1095	0.5202 ±0.0261	0.5300 ±0.0087	0.5973 ±0.0277	0.7660 ±0.0554	0.6303 ±0.0579
	NMI	0.0752 ±0.0206	0.1565 ±0.0365	0.5573 ±0.2763	0.3568 ±0.0381	0.4830 ±0.0079	0.7792 ±0.0143	0.8192 ±0.0165	0.6141 ±0.0874
CESHL	ACC	**0.6450** ±0.0196	**0.4660** ±0.0267	**0.8544** ±0.0505	**0.8020** ±0.0252	**0.5341** ±0.0088	**0.6120** ±0.0121	**0.8250** ±0.0299	**0.6845** ±0.0511
	NMI	**0.0761** ±0.0211	**0.1938** ±0.0317	**0.6793** ±0.1271	**0.5458** ±0.0363	**0.4905** ±0.0046	**0.7880** ±0.019	**0.8476** ±0.0200	**0.6630** ±0.0433

Table 5 shows the results of this ablation study. From Table 5, we find that CESHL-s outperforms CESHL-W on most data sets which demonstrates that the initial quality score is helpful to the ensemble learning. Moreover, CESHL outperforms CESHL-s, which means the propagation process can further improve the performance of clustering ensemble. Therefore, the results demonstrate the effectiveness of our strategy of evaluating the weights of base clusters.

4.5. Parameter study

In this subsection, we show the affect of the hyper-parameter λ. Fig. 4 shows the ACC and NMI results on Arcene and ORL data set. Results on other data sets are similar. We can find that our method is insensitive w.r.t. λ in the range $[10^0, 10^3]$, and thus we can achieve a good performance by setting λ as a value in $[10^0, 10^3]$.

5. Conclusion

In this paper, we proposed a novel clustering ensemble method via structured hypergraph learning. Different from conventional hypergraph based clustering ensemble methods which apply the hypergraph partitioning algorithm on a static hypergraph, the proposed one dynamically learned a structured hypergraph for clustering ensemble. It first evaluated the quality of each base cluster and then learned a structured hypergraph based on such cluster quality. The learned hypergraph had a clear clustering structure, i.e., it contained exact c connective components, and thus we can easily obtain the final consensus clustering results from it. We conducted experiments by comparing with the state-of-the-art clustering ensemble methods on benchmark data sets. The experimental results showed that the proposed method outperformed both the conventional hypergraph based methods and the state-of-the-art clustering ensemble methods, which demonstrated the effectiveness and superiority of the proposed method.

Although the time complexity of CESHL is comparable with other clustering ensemble methods, it is still quadratic in the number of instances. It will limit the applications on the large scale data sets. In the future, we will study how to address this scalable issue by speeding up CESHL.

CRediT authorship contribution statement

Peng Zhou: Conceptualization, Formal analysis, Methodology, Software, Writing – original draft. **Xia Wang:** Methodology, Software. **Liang Du:** Writing – review & editing. **Xuejun Li:** Writing – review & editing, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China grants 62176001, 61806003, 61976129, and 61972001; the Natural Science Foundation of Anhui Province, China grant 1908085MF188; the Key Natural Science Project of Anhui Provincial Education Department, China KJ2020A0041.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.inffus.2021.09.003.

References

[1] A. Strehl, J. Ghosh, Cluster ensembles — A knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. 3 (3) (2003) 583–617.
[2] A. Topchy, A.K. Jain, W.F. Punch, Combining multiple weak clusterings, in: ICDM, 2003, pp. 331–338.
[3] X.Z. Fern, C.E. Brodley, Solving cluster ensemble problems by bipartite graph partitioning, in: ICML, 2004, p. 36.
[4] Z. Zhou, W. Tang, Clusterer ensemble, Knowl. Based Syst. 19 (1) (2006) 77–83.
[5] H. Liu, T. Liu, J. Wu, D. Tao, Y. Fu, Spectral ensemble clustering, in: SIGKDD, 2015, pp. 715–724.
[6] D. Huang, J. Lai, C. Wang, Robust ensemble clustering using probability trajectories, IEEE Trans. Knowl. Data Eng. 28 (5) (2016) 1312–1326.
[7] X. Liu, X. Zhu, M. Li, L. Wang, C. Tang, J. Yin, D. Shen, H. Wang, W. Gao, Late fusion incomplete multi-view clustering, IEEE Trans. Pattern Anal. Mach. Intell. 41 (10) (2019) 2410–2423.
[8] F. Li, Y. Qian, J. Wang, C. Dang, L. Jing, Clustering ensemble based on sample's stability, Artificial Intelligence 273 (2019) 37–55.
[9] L. Bai, J. Liang, F. Cao, A multiple k-means clustering ensemble algorithm to find nonlinearly separable clusters, Inf. Fusion 61 (2020) 36–47.
[10] S. Mimaroglu, E. Erdil, Combining multiple clusterings using similarity graph, Pattern Recognit. 44 (3) (2011) 694–703.
[11] P. Zhou, L. Du, Y.-D. Shen, X. Li, Tri-level robust clustering ensemble with multiple graph learning, in: Thirty-Fifth AAAI Conference on Artificial Intelligence, 2021, pp. 11125–11133.
[12] Z. Tao, H. Liu, Y. Fu, Simultaneous clustering and ensemble, in: AAAI, 2017, pp. 1546–1552.
[13] C. Tang, X. Liu, X. Zhu, J. Xiong, M. Li, J. Xia, X. Wang, L. Wang, Feature selective projection with low-rank embedding and dual Laplacian regularization, IEEE Trans. Knowl. Data Eng. 32 (9) (2020) 1747–1760.
[14] D. Zhou, J. Huang, B. Schölkopf, Learning with hypergraphs: Clustering, classification, and embedding, in: Proceedings of the 19th International Conference on Neural Information Processing Systems, in: NIPS'06, MIT Press, Cambridge, MA, USA, 2006, pp. 1601–1608.
[15] A. Topchy, A.K. Jain, W.F. Punch, A mixture model for clustering ensembles, in: SDM, 2004, pp. 379–390.
[16] J. Yu, Y. Rui, Y.Y. Tang, D. Tao, High-order distance-based multiview stochastic learning in image classification, IEEE Trans. Cybern. 44 (12) (2014) 2431–2442.
[17] P. Zhou, Y.-D. Shen, L. Du, F. Ye, X. Li, Incremental multi-view spectral clustering, Knowl.-Based Syst. 174 (2019) 73–86.
[18] Z. Kang, G. Shi, S. Huang, W. Chen, X. Pu, J.T. Zhou, Z. Xu, Multi-graph fusion for multi-view spectral clustering, Knowl. Based Syst. 189 (2020).
[19] C. Tang, X. Liu, X. Zhu, E. Zhu, Z. Luo, L. Wang, W. Gao, CGD: Multi-view clustering via cross-view graph diffusion, in: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, 2020, pp. 5924–5931.

[20] C. Tang, X. Zheng, X. Liu, W. Zhang, J. Zhang, J. Xiong, L. Wang, Cross-view locality preserved diversity and consensus learning for multi-view unsupervised feature selection, IEEE Trans. Knowl. Data Eng. (2021) 1–12.
[21] A.P. Topchy, A.K. Jain, W.F. Punch, Clustering ensembles: Models of consensus and weak partitions, IEEE Trans. Pattern Anal. Mach. Intell. 27 (12) (2005) 1866–1881.
[22] N. Nguyen, R. Caruana, Consensus clusterings, in: Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007), October 28-31, 2007, Omaha, Nebraska, USA, IEEE Computer Society, 2007, pp. 607–612.
[23] L. Bai, J. Liang, H. Du, Y. Guo, An information-theoretical framework for cluster ensemble, IEEE Trans. Knowl. Data Eng. 31 (8) (2019) 1464–1477.
[24] P. Hore, L.O. Hall, D.B. Goldgof, A scalable framework for cluster ensembles, Pattern Recognit. 42 (5) (2009) 676–688.
[25] F. Li, Y. Qian, J. Wang, J. Liang, Multigranulation information fusion: A Dempster-Shafer evidence theory-based clustering ensemble method, Inform. Sci. 378 (2017) 389–409.
[26] P. Zhou, L. Du, X. Li, Self-paced consensus clustering with bipartite graph, in: C. Bessiere (Ed.), Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, ijcai.org, 2020, pp. 2133–2139.
[27] N. Iam-on, T. Boongoen, S.M. Garrett, C.J. Price, A link-based approach to the cluster ensemble problem, IEEE Trans. Pattern Anal. Mach. Intell. 33 (12) (2011) 2396–2409.
[28] N. Iam-on, T. Boongoen, S.M. Garrett, C.J. Price, A link-based cluster ensemble approach for categorical data clustering, IEEE Trans. Knowl. Data Eng. 24 (3) (2012) 413–425.
[29] Z. Tao, H. Liu, S. Li, Y. Fu, Robust spectral ensemble clustering, in: CIKM, 2016, pp. 367–376.
[30] Z. Tao, H. Liu, S. Li, Z. Ding, Y. Fu, Robust spectral ensemble clustering via rank minimization, ACM Trans. Knowl. Discov. Data 13 (1) (2019) 1–25.
[31] D. Huang, C. Wang, J. Wu, J. Lai, C. Kwoh, Ultra-scalable spectral clustering and ensemble clustering, IEEE Trans. Knowl. Data Eng. 32 (6) (2020) 1212–1226.
[32] Z. Tao, H. Liu, J. Li, Z. Wang, Y. Fu, Adversarial graph embedding for ensemble clustering, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI, 2019, pp. 3562–3568.
[33] G. Karypis, R. Aggarwal, V. Kumar, S. Shekhar, Multilevel hypergraph partitioning: applications in VLSI domain, IEEE Trans. Very Large Scale Integr. Syst. 7 (1) (1999) 69–79.
[34] L. Zhu, J. Shen, L. Xie, Z. Cheng, Unsupervised topic hypergraph hashing for efficient mobile image retrieval, IEEE Trans. Cybern. 47 (11) (2017) 3941–3954.
[35] X. Zhu, Y. Zhu, S. Zhang, R. Hu, W. He, Adaptive hypergraph learning for unsupervised feature selection, in: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI, 2017, pp. 3581–3587.
[36] Z. Zhang, H. Lin, Y. Gao, Dynamic hypergraph structure learning, in: Jérôme Lang(Ed.), Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, 2018, pp. 3162–3169.
[37] W. Zhao, S. Tan, Z. Guan, B. Zhang, M. Gong, Z. Cao, Q. Wang, Learning to map social network users by unified manifold alignment on hypergraph, IEEE Trans. Neural Netw. Learn. Syst. 29 (12) (2018) 5834–5846.
[38] P. Purkait, T. Chin, A. Sadri, D. Suter, Clustering with hypergraphs: The case for large hyperedges, IEEE Trans. Pattern Anal. Mach. Intell. 39 (9) (2017) 1697–1711.
[39] Y. Gao, M. Wang, D. Tao, R. Ji, Q. Dai, 3-D object retrieval and recognition with hypergraph analysis, IEEE Trans. Image Process. 21 (9) (2012) 4290–4303.
[40] J. Yu, D. Tao, M. Wang, Adaptive hypergraph learning and its application in image classification, IEEE Trans. Image Process. 21 (7) (2012) 3262–3272.
[41] J. Yu, Y. Rui, B. Chen, Exploiting click constraints and multi-view features for image re-ranking, IEEE Trans. Multimed. 16 (1) (2014) 159–168.
[42] C. Tang, X. Liu, P. Wang, C. Zhang, M. Li, L. Wang, Adaptive hypergraph embedded semi-supervised multi-label image annotation, IEEE Trans. Multimed. 21 (11) (2019) 2837–2849.
[43] D. Zhou, J. Weston, A. Gretton, O. Bousquet, B. Schölkopf, Ranking on data manifolds, in: Advances in Neural Information Processing Systems, Vol. 16, NIPS 2003, MIT Press, 2003, pp. 169–176.
[44] K. Fan, On a theorem of weyl concerning eigenvalues of linear transformations: Ii*, Proc. Natl. Acad. Sci. USA 36 (1) (1949) 31–35.
[45] D.C. Liu, J. Nocedal, On the limited memory BFGS method for large scale optimization, Math. Program. 45 (1–3) (1989) 503–528.
[46] P. Zhou, L. Du, H. Wang, L. Shi, Y. Shen, Learning a robust consensus matrix for clustering ensemble via Kullback-Leibler divergence minimization, in: IJCAI, 2015, pp. 4112–4118.
[47] J. Zhou, H. Zheng, L. Pan, Ensemble clustering based on dense representation, Neurocomputing 357 (2019) 66–76.
[48] P. Zhou, L. Du, X. Liu, Y. Shen, M. Fan, X. Li, Self-paced clustering ensemble, IEEE Trans. Neural Netw. Learn. Syst. 32 (4) (2021) 1497–1511.
[49] Y. Zhao, G. Karypis, Empirical and theoretical comparisons of selected criterion functions for document clustering, Mach. Learn. 55 (3) (2004) 311–331.
[50] T. Li, C.H.Q. Ding, Weighted consensus clustering, in: SDM, 2008, pp. 798–809.
[51] D. Huang, C. Wang, J. Lai, Locally weighted ensemble clustering, IEEE Trans. Syst. Man Cybern. 48 (5) (2018) 1460–1473.