Dynamic Anchor-based Ensemble Clustering via Hypergraph Reconstruction

Jiaxuan Xu[1], **Lei Duan**[1*], **Xinye Wang**[1] and **Liang Du**[2]
[1]School of Computer Science, Sichuan University, Chengdu, China
[2]School of Computer and Information Technology, Shanxi University, Taiyuan, China
xujx@stu.scu.edu.cn, leiduan@scu.cn, wangxinye@stu.scu.edu.cn, duliang@sxu.edu.cn

Abstract

Ensemble clustering learns a consensus result by integrating a set of base clustering results. Recently, anchor-based methods construct an anchor similarity matrix to represent the affinity relationships among samples, significantly improving computational efficiency. However, these methods struggle with fixed anchors generated by static anchor learning strategies, which lead to low-quality anchor similarity matrix and poor clustering accuracy. To address this issue, we propose a novel method named d**Y**namic **A**nchor-based ensemble **C**lustering via **H**ypergraph reconstruc**T**ion (YACHT). Specifically, YACHT first transforms the base clustering results into a hypergraph and designs a novel hypergraph enhancement strategy to improve the reliability of the initial hypergraph. YACHT reconstructs the hypergraph through matrix factorization and introduces a mapping matrix to filter out redundant information, capturing a high-quality anchor similarity matrix. Then, YACHT attempts to incorporate the hypergraph into the optimization objective to achieve hypergraph updates. To ensure the accuracy of hypergraph updates, we impose a hypergraph regularizer and a local consensus information alignment term. The alignment term is implemented by minimizing the discrepancy between the label partition derived from the hypergraph regularizer and the local consensus information indicator matrix extracted from the base clustering results. Extensive experimental results demonstrate the outstanding performance of the proposed YACHT. The code is available at https://github.com/scu-kdde/YACHT.

1 Introduction

Ensemble clustering integrates multiple base clustering results to capture a more robust consensus result [Chen *et al.*, 2023; Shi *et al.*, 2021; Li *et al.*, 2021; Bai *et al.*, 2020]. Without accessing the data features, the range and level of effective information that ensemble clustering can utilize are quite limited, which also gives it the characteristic of protecting data privacy [Zhou *et al.*, 2024]. Most existing ensemble clustering methods focus on converting base clustering results into a Co-association (CA) matrix. The CA matrix describes the frequency with which samples belong to the same cluster and characterizes the cluster structure at the sample level. These methods incorporate various learning paradigms or graph-based techniques to improve the quality of the CA matrix, ultimately obtaining discrete consensus clustering result through hierarchical or spectral clustering methods [Jia *et al.*, 2021; Xu *et al.*, 2024]. However, the $n \times n$-sized CA matrix inevitably leads to a time complexity of $O\left(n^2\right)$ or even $O\left(n^3\right)$, making it difficult to scale to large-scale datasets.

Anchor-based ensemble clustering methods are proposed to address this scalability issue. These methods achieve the ensemble by selecting representative samples (anchors), learning the relationships between the anchors and the samples (the anchor similarity matrix), and clustering in three steps [Zhang *et al.*, 2024a; Liu *et al.*, 2024]. The strategy for selecting anchors is crucial for the quality of the anchor similarity matrix and directly impacts clustering accuracy. Existing methods treat the cluster centers obtained by K-means and its variants as anchors, and once the anchors are selected, they remain fixed [Huang *et al.*, 2020; Zhang *et al.*, 2024a; Li *et al.*, 2023]. This reduces the flexibility and quality of the constructed anchor similarity matrix, and in practical applications, the number of anchors is often large. Moreover, these methods use data features when selecting anchors, which limits the privacy-preserving capabilities of ensemble clustering. To address these issues, there is an urgent need to propose a method that dynamically learns anchors without relying on data features, in order to preserve the privacy-preserving nature of ensemble clustering and improve the quality of the anchor similarity matrix. Specifically, this method should meet the following challenges. ($\mathbf{C1}$): How to improve the reliability of base clustering results to enhance the accuracy of subsequent anchor point learning. ($\mathbf{C2}$): How to dynamically learn anchor points and construct high-quality anchor similarity matrices using only base clustering results to preserve the privacy-preserving characteristics of ensemble clustering. ($\mathbf{C3}$): How to fully leverage the effective local information in base clustering results to improve clustering accuracy.

To address these challenges, we propose a novel method named dynamic anchor-based ensemble clustering via hyper-

*Corresponding Author

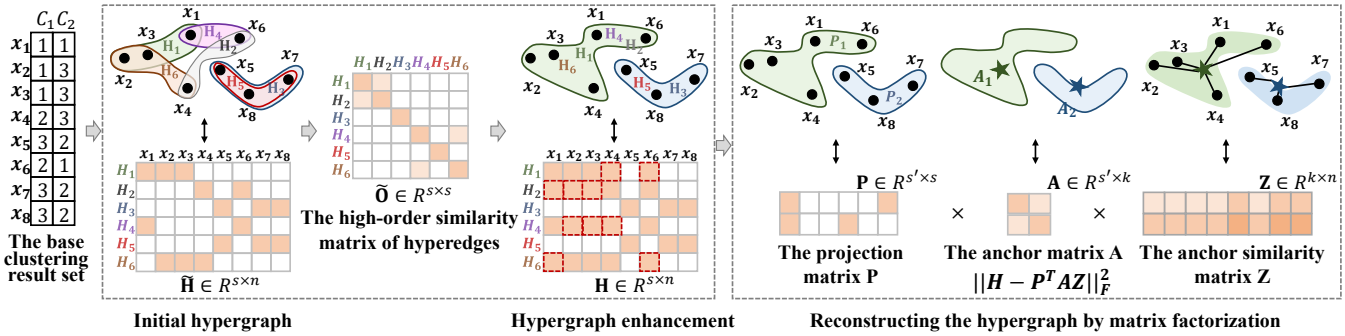


Figure 1: We construct an initial hypergraph $\widetilde{\mathbf{H}}$ and then perform random walks on the hyperedges to enhance the initial hypergraph.

graph reconstruction (YACHT). Specifically, as shown in Figure 1, YACHT transforms the base clustering results into a hypergraph and designs a hypergraph enhancement strategy based on random walks on hyperedges to improve the reliability of the initial hypergraph. Then, YACHT reconstructs the hypergraph through matrix factorization. During the reconstruction process, both the anchor matrix and the anchor similarity matrix are incorporated into the optimization objective for dynamic learning. Moreover, a mapping matrix is introduced to filter out redundant information in the hypergraph. This helps capture a higher-quality anchor similarity matrix. Next, YACHT incorporates the hypergraph into the optimization objective and imposes a hypergraph regularizer. The regularizer constrains the hypergraph update and derives a label partition. YACHT aligns the label partition with the local consensus information extracted from the base clustering results. This alignment strategy helps constrain the hypergraph to retain local consensus information during the update process without introducing significant deviations. Finally, the final consensus results are obtained by applying the K-means method to the anchor similarity matrix.

The main contributions of this paper are:

- We propose a novel method named dynamic anchor-based ensemble clustering via hypergraph reconstruction (YACHT). Compared to methods based on the CA matrix, YACHT reduces the time complexity to $O(n)$ and is more suitable for large-scale datasets.
- We introduce a hypergraph enhancement strategy to improve the reliability of the initial hypergraph. The hypergraph regularizer and local consensus information alignment term are imposed to prevent significant deviations during the hypergraph update process.
- We design an alternating optimization strategy for the objective function. The effectiveness of YACHT is demonstrated through comparisons with 17 representative clustering methods.

2 Related Work

2.1 Ensemble Clustering

Strehl and Ghosh [Strehl and Ghosh, 2003] conducted early research on ensemble clustering, defining ensemble clustering as the process of learning a consensus result from a set of base clustering results. Ensemble clustering alleviates the robustness and stability issues of single-clustering methods. It can be broadly divided into two categories: methods based on the quality of base clustering results and methods based on consensus function strategies. The former either selects high-quality base clustering results from the base clustering set or relies on robust clustering algorithms to generate more accurate base clustering results [Huang *et al.*, 2023; Abbasi *et al.*, 2019; Zhao *et al.*, 2017]. Methods based on consensus function strategies have garnered wide attention due to their excellent performance. These include CA matrix-based methods [Tao *et al.*, 2019; Liu *et al.*, 2015; Xie *et al.*, 2024] and bipartite graph-based methods. CA matrix-based methods focus on improving the quality of the CA matrix [Hao *et al.*, 2024; Jia *et al.*, 2024] or learning its underlying subspace to explore cluster structures [Tao *et al.*, 2019; Tao *et al.*, 2021]. Bipartite graph-based methods leverage the association between samples and clusters to obtain consensus partitions [Fern and Brodley, 2004]. Some of these methods adopt self-paced learning frameworks to progressively learn consensus results [Zhou *et al.*, 2021a].

2.2 Anchor based Clustering

A large number of anchor-based clustering methods have been proposed [Liu *et al.*, 2024; Wan *et al.*, 2023]. The motivation for introducing anchors in clustering is to enhance the scalability of the algorithm. Anchors are representative samples selected from the data, which are then used to learn the affinity relationships among the samples. Research on anchors in ensemble clustering is still in its early stages. Representative methods include: Yang *et al.* performed spectral clustering based on anchors and then assigned samples to the nearest anchor clusters, significantly improving timeliness [Yang *et al.*, 2023]. Li *et al.* proposed an ensemble method that reuses anchor similarity matrix and employs light-k-means [Li *et al.*, 2023]. Zhang *et al.* introduced a fast k-nearest neighbor approximation method to construct similarity matrices for approximating affinity matrix [Zhang *et al.*, 2024a]. However, a key limitation of these methods is that they used raw data features to select anchors, compromising the privacy-protecting characteristics of ensemble clustering. Huang and Liang's methods [Huang *et al.*, 2020; Liang *et al.*, 2020] did not incorporate data features. However, their methods either used fixed anchors or ignore local

consensus information, leading to poorer performance. Unlike the methods mentioned above, our method does not introduce data features, allowing for dynamic anchor learning and fully leveraging local consensus information.

3 The Proposed Method

This section details the YACHT method, covering the objective function, optimization, and time complexity analysis.

3.1 Constructing and Enhancing the Hypergraph

Given n samples $\mathcal{X} = \{x_1, x_2, ..., x_n\}$ and m base clustering results $\Xi = \{C_1, C_2, ..., C_m\}$, where x_i denotes the i-th sample. $C_j = \{\pi_j^1, \pi_j^2, ..., \pi_j^{s_j}\}$ represents the j-th base clustering result that partitions $\mathcal{X}$ into s_j base clusters. π_j^h represents the h-th base cluster in C_j. A hypergraph $\mathcal{G} = \left(\mathcal{V}, \mathcal{E}, \widetilde{\mathbf{W}}\right)$ can be constructed from the base clustering results, where $\mathcal{V}$ represents the set of nodes, each corresponding to a sample. Each hyperedge in the hyperedge set $\mathcal{E}$ can represent a base cluster. Therefore, the number of hyperedges is equal to the number of base clusters $s = \sum_{j=1}^{m} s_j$. As shown in Figure 1, we can construct the initial hypergraph indicator matrix $\widetilde{\mathbf{H}}$. The hyperedge weight $\widetilde{\mathbf{W}}$ is a diagonal matrix, where the diagonal elements $\widetilde{\mathbf{W}}_{ii}$ represents the weights of the i-th hyperedge. Since each hyperedge represents a base cluster, we use local entropy weights [Huang *et al.*, 2018] to represent the weights of the hyperedges. The hyperedge weights are defined as:

$$\widetilde{\mathbf{W}}_{ii} = e^{\frac{\sum_{j=1}^{s} p\left(\widetilde{\mathbf{H}}_{i:}, \widetilde{\mathbf{H}}_{j:}\right) log_2 p\left(\widetilde{\mathbf{H}}_{i:}, \widetilde{\mathbf{H}}_{j:}\right)}{\theta \cdot m}},$$

where $p(\widetilde{\mathbf{H}}_{i:}, \widetilde{\mathbf{H}}_{j:}) = \frac{|\widetilde{\mathbf{H}}_{i:} \bigcap \widetilde{\mathbf{H}}_{j:}|}{|\widetilde{\mathbf{H}}_{i:}|}$ represents the ratio of the number of intersecting samples between two hyperedges to the number of samples in i-th hyperedge. θ represents the weight decay parameter, where a smaller value indicates faster weight decay.

Due to the weaknesses of the base clustering algorithms, the reliability and stability of the initial hypergraph generated from the base clustering results are compromised [Zhou *et al.*, 2022]. We improve the reliability of the hypergraph by introducing random walks to explore the proximity between hyperedges [Zhang *et al.*, 2024b]. We use the Jaccard distance to define the direct similarity between hyperedges:

$$\mathbf{O}_{ij} = \text{Jaccard}(\pi^i, \pi^j), \quad \text{Jaccard}(\pi^i, \pi^j) = \frac{|\pi^i \cap \pi^j|}{|\pi^i \cup \pi^j|},$$

where $\mathbf{O}_{ij}$ represents the similarity between the i-th and j-th hyperedges. Normalizing $\mathbf{O}$ as $\dot{\mathbf{O}} = \mathbb{D}_{\mathbf{o}}^{-1}\mathbf{O}$ allows it to be interpreted as a probability transition matrix, where $\mathbb{D}_{\mathbf{o}}$ denotes the Laplacian matrix of $\mathbf{O}$. Then, the high-order relationship matrix $\widetilde{\mathbf{O}} \in \mathbb{R}^{s \times s}$ of the hyperedges is defined as:

$$\widetilde{\mathbf{O}} = \dot{\mathbf{O}}^{(1)\top}\dot{\mathbf{O}} + \dot{\mathbf{O}}^{(2)\top}\dot{\mathbf{O}} + \cdots + \dot{\mathbf{O}}^{(d)\top}\dot{\mathbf{O}}, \tag{1}$$

where $\dot{\mathbf{O}}^{(d)} = \dot{\mathbf{O}}^{(d-1)\top}\dot{\mathbf{O}}$. Based on this similarity, we can improve the existing hypergraph structure. We generate the enhanced hypergraph $\mathbf{H} \in \mathbb{R}^{s \times n}$ as follows:

$$\mathbf{H}_{i:} = \begin{cases} \mathbf{1}\left(\sum_{j \in \{j | \widetilde{\mathbf{O}}_{ij} \geq \alpha\}} \widetilde{\mathbf{H}}_{j:}\right) & \text{if } \sum \sum_{j \in \{j | \widetilde{\mathbf{O}}_{ij} \geq \alpha\}} \widetilde{\mathbf{H}}_{j:} > 0, \\ \widetilde{\mathbf{H}}_{i:} & \text{otherwise,} \end{cases} \tag{2}$$

where $\mathbf{1}(\cdot)$ is a mapping function that maps all values greater than 0 to 1, and α is the similarity threshold. As shown in Eq. (2), the hyperedges of the enhanced hypergraph $\mathbf{H}$ may change, resulting in new hyperedge weights $\mathbf{W}$.

3.2 Reconstructing the Hypergraph with Hypergraph Regularization

After obtaining the enhanced hypergraph $\mathbf{H}$, a self-expressive model is employed to capture sample affinities:

$$\min_{\mathbf{S}} \|\mathbf{H} - \mathbf{DS}\|_F^2,$$

where $\mathbf{S} \in \mathbb{R}^{n \times n}$ denotes the similarity matrix and $\mathbf{D}$ is the dictionary. This form can reconstruct $\mathbf{H}$, but has the following issues: (i) the similarity matrix $\mathbf{S}$ is $n \times n$, and the reconstruction of $\mathbf{H}$ is slow, leading to lower computational efficiency; (ii) it cannot filter out the redundant information in the hypergraph. Therefore, we introduce anchor learning and a mapping matrix. The anchor points reduce the similarity matrix dimension from $n \times n$ to $k \times n$ relations, where k is the number of anchor points. This results in a smaller anchor similarity matrix $\mathbf{Z}$ compared to $\mathbf{S}$, thereby accelerating the reconstruction process. The introduction of the mapping matrix helps filter out redundant hyperedges in the hypergraph, which aids in capturing a higher-quality anchor similarity matrix. Therefore, we obtain the following form:

$$\begin{aligned} &\min_{\mathbf{P},\mathbf{A},\mathbf{Z}} \|\mathbf{H} - \mathbf{P}^\top \mathbf{AZ}\|_F^2 \\ &\text{s.t. } \mathbf{PP}^\top = \mathbf{I}, \mathbf{AA}^\top = \mathbf{I}, \mathbf{Z}^\top \mathbf{1} = \mathbf{1}, \mathbf{Z}_{ij} \geq 0, \end{aligned} \tag{3}$$

where $\mathbf{P} \in \mathbb{R}^{s' \times s}$ is the projection matrix, $\mathbf{Z} \in \mathbb{R}^{k \times n}$ is the similarity matrix representing the relationship between the anchors and the samples, and $\mathbf{A} \in \mathbb{R}^{s' \times k}$ denotes the anchor matrix. s' is the number of representative base clusters. The orthogonal constraints on variables $\mathbf{P}$ and $\mathbf{A}$ help learn a better similarity matrix $\mathbf{Z}$ [Feng *et al.*, 2024].

Furthermore, we attempt to incorporate $\mathbf{H}$ into the optimization objective, as the quality of $\mathbf{H}$ directly impacts the reliability of $\mathbf{Z}$. However, unconstrained updates inevitably lead to large errors and unreliable $\mathbf{Z}$. Therefore, to learn a better hypergraph, we apply the widely used hypergraph regularizer [Gao *et al.*, 2022; Zhang *et al.*, 2018]. We rewrite equation (3) as:

$$\begin{aligned} &\min_{\mathbf{H},\mathbf{P},\mathbf{A},\mathbf{Z},\mathbf{F}} \|\mathbf{H} - \mathbf{P}^\top \mathbf{AZ}\|_F^2 + \mathbf{tr}\left(\mathbf{F}^\top \mathbf{LF}\right) \\ &\text{s.t. } \mathbf{PP}^\top = \mathbf{I}, \mathbf{AA}^\top = \mathbf{I}, \mathbf{FF}^\top = \mathbf{I}, \mathbf{Z}^\top \mathbf{1} = \mathbf{1}, \mathbf{Z}_{ij} \geq 0, \end{aligned} \tag{4}$$

where $\mathbf{L} = \mathbf{I} - \mathbf{D}_v^{-1/2}\mathbf{H}^\top \mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}\mathbf{D}_v^{-1/2}$ is the Laplacian matrix of the hypergraph. $\mathbf{D}_v = diag(\mathbf{H}^\top \mathbf{W1})$ denotes the node degree matrix, $\mathbf{D}_e = diag(\mathbf{1}^\top \mathbf{H}^\top)$ is the edge degree matrix. This regularizer not only constrains the hypergraph update but also derives a label indicator matrix $\mathbf{F} \in \mathbb{R}^{n \times c}$, where c is the number of clusters. The label indicator matrix helps us align the local consensus information derived

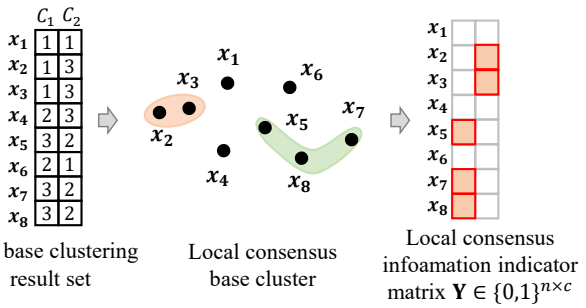


Figure 2: x_2 and x_3 form a consensus cluster, as do x_5, x_7 and x_8. The consensus clusters are ranked by size, and the top c clusters are selected to generate the $\mathbf{Y}$ matrix. c denotes the number of clusters.

from the base clustering results. This ensures that the hypergraph preserves this information during the optimization process while avoiding significant deviations.

3.3 Aligning Local Consensus Information

In general, local consensus information refers to results that are reflected in all base clustering results, for example, when two samples are clustered together in every base clustering perspective. Since the CA matrix captures relationships at the sample level, local consensus information can be easily applied in methods based on the CA matrix [Jia *et al.*, 2021; Jia *et al.*, 2024; Li *et al.*, 2024]. To align local consensus information with the label indicator matrix $\mathbf{F}$ derived from the hypergraph, we also need to define a consensus base cluster S to match the size of $\mathbf{F}$.

Definition 1. *Let S be a set of objects. The set S is a consensus base cluster if and only if (i) $\forall x_i, x_j \in S$ s.t.$\widetilde{\mathbf{H}}_{:i} = \widetilde{\mathbf{H}}_{:j}$, and (ii) the number of samples contained in the set, $num\,(S)$, is at least 2 $(num\,(S))$.*

Definition 1 provides the definition of consensus base cluster based on the initial hypergraph. Although both consensus base clusters and microclusters [Huang *et al.*, 2016] require consistent sample-base cluster relationships (i.e. $\widetilde{\mathbf{H}}_{:i} = \widetilde{\mathbf{H}}_{:j}$), we additionally require that consensus base clusters contain more than 2 samples. This is because microclusters can include single samples, but retaining a consensus base cluster with only a single sample for alignment is meaningless. Considering the size of $\mathbf{F}$, we strive to retain as much information as possible from the large number of local consensus base clusters. We first sort the number of samples in all consensus base clusters and then retain the top c consensus base clusters. Based on these c base clusters, we generate the local consensus information indicator matrix. Figure 2 shows the construction process of the local consensus information indicator matrix $\mathbf{Y}$. Note that all of this is based solely on the base clustering results, without relying on any ground truth. Finally, we minimize the error between $\mathbf{F}$ and $\mathbf{Y}$ to achieve alignment. The final objective function is as follows:

$$\begin{aligned} &\min_{\mathbf{H},\mathbf{P},\mathbf{A},\mathbf{Z},\mathbf{F}} \|\mathbf{H} - \mathbf{P}^\top \mathbf{A}\mathbf{Z}\|_F^2 + \mathbf{tr}\left(\mathbf{F}^\top \mathbf{L}\mathbf{F}\right) + \|\mathbf{F} - \mathbf{Y}\|_F^2 \\ &\text{s.t. } \mathbf{P}\mathbf{P}^\top = \mathbf{I}, \mathbf{A}\mathbf{A}^\top = \mathbf{I}, \mathbf{F}\mathbf{F}^\top = \mathbf{I}, \mathbf{Z}^\top \mathbf{1} = \mathbf{1}, \mathbf{Z}_{ij} \geq 0. \end{aligned} \tag{5}$$

3.4 Optimization

To solve Eq. (5), we employ the alternating optimization strategy [Zhong and Pun, 2022], i.e., optimizing the target variable alternately while fixing other variables.

Update P To update the variable $\mathbf{P}$, other variables need to be fixed. The subproblem w.r.t. $\mathbf{P}$ is as follows:

$$\min_{\mathbf{P}} \|\mathbf{H} - \mathbf{P}^\top \mathbf{A}\mathbf{Z}\|_F^2 \quad \text{s.t. } \mathbf{P}\mathbf{P}^\top = \mathbf{I}. \tag{6}$$

We expand the Frobenius norm by the trace operation and remove terms unrelated to $\mathbf{P}$, further transforming Eq. (6) into the following form:

$$\max_{\mathbf{P}} \mathbf{tr}(\mathbf{P}^\top \mathbf{Q}) \quad \text{s.t. } \mathbf{P}\mathbf{P}^\top = \mathbf{I}, \tag{7}$$

where $\mathbf{Q} = \mathbf{A}\mathbf{Z}\mathbf{H}^\top$. Supposing the Singular value decomposition (SVD) of $\mathbf{Q}$ is $\mathbf{U}_\mathbf{Q}\Sigma_\mathbf{Q}\mathbf{V}_\mathbf{Q}^\top$, we can obtain the optimal solution for $\mathbf{P}$ as $\mathbf{U}_\mathbf{Q}\mathbf{V}_\mathbf{Q}^\top$.

Update A With $\mathbf{P}$, $\mathbf{Z}$, $\mathbf{H}$ and $\mathbf{F}$ being fixed, the subproblem for $\mathbf{A}$ is as follows:

$$\min_{\mathbf{A}} \|\mathbf{H} - \mathbf{P}^\top \mathbf{A}\mathbf{Z}\|_F^2 \quad \text{s.t. } \mathbf{A}\mathbf{A}^\top = \mathbf{I}.$$

Similar to the subproblem for $\mathbf{P}$, we transform the subproblem for $\mathbf{A}$ into the following form by expanding the Frobenius norm:

$$\max_{\mathbf{A}} \mathbf{tr}(\mathbf{A}^\top \mathbf{B}) \quad \text{s.t. } \mathbf{A}\mathbf{A}^\top = \mathbf{I}, \tag{8}$$

where $\mathbf{B} = \mathbf{P}\mathbf{H}\mathbf{Z}^\top$. Similarly, we perform the SVD decomposition $\mathbf{B} = \mathbf{U}_\mathbf{B}\Sigma_\mathbf{B}\mathbf{V}_\mathbf{B}^\top$, and then obtain the optimal solution for the variable $\mathbf{A}$ as $\mathbf{U}_\mathbf{B}\mathbf{V}_\mathbf{B}^\top$.

Update Z With $\mathbf{P}$, $\mathbf{A}$, $\mathbf{H}$ and $\mathbf{F}$ being fixed, the subproblem for $\mathbf{Z}$ is as follows:

$$\min_{\mathbf{Z}} \|\mathbf{H} - \mathbf{P}^\top \mathbf{A}\mathbf{Z}\|_F^2 \quad \text{s.t. } \mathbf{Z}^\top \mathbf{1} = \mathbf{1}, \mathbf{Z}_{ij} \geq 0. \tag{9}$$

Eq. (9) can be solved using the augmented Lagrange multiplier (ALM) method [Bazaraa *et al.*, 2013]. Its augmented Lagrangian function is as follows:

$$\mathcal{L}(\mathbf{Z}) = \|\mathbf{H} - \mathbf{P}^\top \mathbf{A}\mathbf{Z}\|_F^2 + \mathbf{J}^\top \left(\mathbf{Z}^\top \mathbf{1} - \mathbf{1}\right) + \frac{\mu}{2}\|\mathbf{Z}^\top \mathbf{1} - \mathbf{1}\|_F^2, \tag{10}$$

where $\mathbf{J} \in \mathbb{R}^{n \times 1}$ represents the Lagrange multiplier, μ is the penalty parameter. Next, we take the derivative of Eq. (10) and set it to zero. The variable $\mathbf{Z}$ can be updated by:

$$\mathbf{Z} = \left(2\mathbf{A}^\top \mathbf{P}\mathbf{P}^\top \mathbf{A} + \mu \mathbf{1}\mathbf{1}^\top\right)^{-1} \left(\mu \mathbf{1}\mathbf{1}^\top + 2\mathbf{A}^\top \mathbf{P}\mathbf{H} - \mathbf{1}\mathbf{J}^\top\right).$$

Update H With $\mathbf{P}$, $\mathbf{A}$, $\mathbf{Z}$ and $\mathbf{F}$ being fixed, the subproblem for $\mathbf{H}$ is as follows:

$$\min_{\mathbf{H}} \|\mathbf{H} - \mathbf{P}^\top \mathbf{A}\mathbf{Z}\|_F^2 + \mathbf{tr}\left(\mathbf{F}^\top \mathbf{L}\mathbf{F}\right). \tag{11}$$

Since $\mathbf{D}_v$ and $\mathbf{D}_e$ are related to $\mathbf{H}$, we need to first take the derivatives of $\mathbf{D}_v$ and $\mathbf{D}_e$ before taking the derivative of Eq. (11). Denote that $\mathbf{D}_v^{-\frac{1}{2}} = diag\left([d_1^v, ..., d_n^v]\right)$ and $\mathbf{D}_e^{-1} = diag\left([d_1^e, ..., d_n^e]\right)$. We can get:

$$\begin{aligned} \frac{\partial d_i^v}{\partial H_{pq}} &= \frac{\partial \left(\sum_{j=1}^s H_{ji} w_j\right)^{-\frac{1}{2}}}{\partial H_{pq}} = -\frac{1}{2}\left(d_i^v\right)^3 \delta_{iq} w_p, \\ \frac{\partial d_j^e}{\partial H_{pq}} &= \frac{\partial \left(\sum_{i=1}^n H_{ji}\right)^{-1}}{\partial H_{pq}} = -\left(d_j^e\right)^2 \delta_{jp}, \end{aligned}$$

Algorithm 1 Obtain label indicator matrix $\mathbf{F}$

Input: Matrix $\mathbf{G}$, $\mathbf{Y}$.
Output: $\mathbf{F}$.

Initialize an orthogonal matrix $\mathbf{F}$ satisfying $\mathbf{F}\mathbf{F}^\top = \mathbf{I}$.
while not converged **do**
  Update $\mathbf{Q}'$ by $\mathbf{Q}' = \mathbf{G}\mathbf{F} + 2\mathbf{Y}$.
  Calculate the SVD of $\mathbf{Q}'$ = $\mathbf{U}_{\mathbf{Q}'}\Sigma_{\mathbf{Q}'}\mathbf{V}_{\mathbf{Q}'}^\top$.
  Update $\mathbf{F}$ by $\mathbf{F} = \mathbf{U}_{\mathbf{Q}'}\mathbf{V}_{\mathbf{Q}'}^\top$.
end while

where δ is the Kronecker delta, i.e., $\delta_{iq} = 1$ if $i = q$ and $\delta_{iq} = 0$ otherwise. Let the subproblem for Eq. (11) be denoted as $\mathcal{Q}(\mathbf{H})$. The derivative w.r.t. $\mathbf{H}$ can be written as:

$$\begin{aligned}\nabla\mathcal{Q}(\mathbf{H}) = 2(\mathbf{H} - \mathbf{P}^\top\mathbf{A}\mathbf{Z}) - 2\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}\mathbf{D}_v^{-\frac{1}{2}}\mathbf{F}\mathbf{F}^\top\mathbf{D}_v^{-\frac{1}{2}} \\ + \mathbf{D}_v^{-\frac{3}{2}}\operatorname{diag}\left(\mathbf{H}^\top\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}\mathbf{D}_v^{-\frac{1}{2}}\mathbf{F}\mathbf{F}^\top\right)\mathbf{J}\mathbf{W} \\ + \mathbf{J}\operatorname{diag}\left(\mathbf{H}\mathbf{D}_v^{-\frac{1}{2}}\mathbf{F}\mathbf{F}^\top\mathbf{D}_v^{-\frac{1}{2}}\mathbf{H}^\top\mathbf{W}\mathbf{D}_e^{-2}\right),\end{aligned}$$

where $\mathbf{J}$ is a matrix where all elements are 1. Setting the derivative $\nabla\mathcal{Q}$ to 0 and then solving is quite difficult, so we use the projected gradient method to optimize the variable $\mathbf{H}$ [Gao *et al.*, 2022]. $\mathbf{H}$ is updated as follows:

$$\mathbf{H}_{k+1} = \mathbb{P}\left[\mathbf{H}_k - \alpha\nabla\mathcal{Q}(\mathbf{H}_k)\right], \tag{12}$$

where α is the learning rate, and $\mathbb{P}$ is the projection onto the feasible set $\{\mathbf{H}|1 \geq \mathbf{H} \geq 0\}$. $\mathbb{P}$ is defined as follows:

$$\mathbb{P}[\mathbf{H}_{ij}] = \begin{cases}\mathbf{H}_{ij} & \text{if } 0 \leq \mathbf{H}_{ij} \leq 1 \\ 0 & \text{if } \mathbf{H}_{ij} < 0 \\ 1 & \text{if } \mathbf{H}_{ij} > 1\end{cases}. \tag{13}$$

Update F With $\mathbf{P}$, $\mathbf{A}$, $\mathbf{Z}$ and $\mathbf{H}$ being fixed, the subproblem for $\mathbf{F}$ is as follows:

$$\begin{aligned}&\min_{\mathbf{F}}\mathbf{tr}\left(\mathbf{F}^\top\mathbf{L}\mathbf{F}\right) + \|\mathbf{F} - \mathbf{Y}\|_F^2 \quad \text{s.t.}\mathbf{F}\mathbf{F}^\top = \mathbf{I}, \\ &\Rightarrow \max_{\mathbf{F}}\operatorname{tr}(\mathbf{F}^\top\mathbf{G}\mathbf{F}) + 2\operatorname{tr}(\mathbf{F}^\top\mathbf{Y}) \quad \text{s.t.}\mathbf{F}\mathbf{F}^\top = \mathbf{I},\end{aligned} \tag{14}$$

where $\mathbf{G} = \mathbf{D}_v^{-1/2}\mathbf{H}^\top\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}\mathbf{D}_v^{-1/2}$. Inspired by [Zhuge *et al.*, 2019], we solve for variable $\mathbf{F}$ using Algorithm 1 and obtain its closed-form solution.

Update J and μ The Lagrange multipliers and penalty parameters in the ALM method are updated as follows:

$$\mathbf{J} = \mathbf{J} + \mu\left(\mathbf{Z}^\top\mathbf{1} - \mathbf{1}\right),\ \mu = \min\left(\rho\mu, \mu_{max}\right), \tag{15}$$

where $\rho > 0$ is a user-defined constant, and μ_{max} represents the maximum value of μ. The overall process of the algorithm is summarized in Algorithm 2.

3.5 Complexity Analysis

In YACHT, the time complexity is mainly driven by the hypergraph enhancement strategy and the updates of several variables. The hypergraph enhancement strategy involves random walks on the hyperedges and matrix multiplication, with a time complexity of $O(s^3)$. For variable $\mathbf{P}$, its update process involves SVD decomposition and matrix multiplication, resulting in a time complexity of $O(s's^2 + s'n(k+s))$. For variable $\mathbf{A}$, its time complexity is $O(s'k^2 + s'n(k+s))$. For variable $\mathbf{Z}$, it involves matrix inversion, resulting in a time complexity of $O(k^3)$. For variable $\mathbf{H}$, we use the projected gradient method for optimization. In each iteration, the time complexity of the computation can be reduced to $O(n)$ by appropriately applying the associative law of matrix multiplication. Similarly, for variable $\mathbf{F}$, by adjusting the order of matrix multiplication, its computational complexity can be controlled within $O(c^2n)$. According to [Zhou *et al.*, 2022], the number of base clusters is much smaller than the number of samples, so we have $n >> s$, $n >> k$, and $n >> s'$. Overall, the total time complexity of YACHT is linearly related to the number of samples, $O(n)$.

Algorithm 2 YACHT

Input: m base clustering results, c, k and α.
Output: Consensus result $\mathbf{s}$.

Initialize $\mathbf{P}$, $\mathbf{A}$, $\mathbf{Z}$, and $\mathbf{J}$. Construct the initial hypergraph $\widetilde{\mathbf{H}}$ and obtain the enhanced hypergraph $\mathbf{H}$.
while not converged **do**
  Update $\mathbf{P}$ by solving Eq. (7).
  Update $\mathbf{A}$ by solving Eq. (8).
  Update $\mathbf{Z}$ by solving Eq. (9).
  Update $\mathbf{H}$ by solving Eq. (11).
  Update $\mathbf{F}$ by Algorithm 1.
  Update $\mathbf{J}$ and μ by Eq. (15).
end while
Obtain the left singular vector $\mathbf{U}$ by performing SVD on $\mathbf{Z}$. Perform k-means method on $\mathbf{U}$ to obtain $\mathbf{s}$.

No.	Dataset	Instance	Feature	Class
D1	Tr12	313	5804	8
D2	Cars	392	8	3
D3	IS	2310	19	7
D4	Segment	2310	18	7
D5	MnistData_05	3495	653	10
D6	MnistData_10	6996	688	10
D7	COIL100	7200	1024	100
D8	FashionMNIST	60000	784	10
D9	KannadaMNIST	60000	784	10
D10	EMNIST	280000	784	10

Table 1: Characteristics of all datasets.

4 Experiment

4.1 Experimental Setup

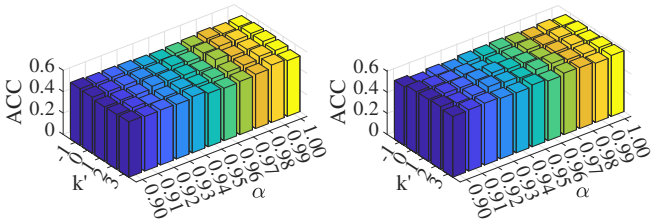
We conduct extensive experiments on 10 real datasets. Characteristics of these datasets are provided in Table 1. Some methods [Zhou *et al.*, 2019; Tao *et al.*, 2019] provide relevant base clustering results, which we use directly for convenience. For the other datasets[1], we randomly run the K-means method 100 times to generate the base clustering result set.

[1]http://archive.ics.uci.edu/datasets

Dataset	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
					ACC					
Base clusterings (average)	45.2±5.7	23.0±8.6	38.7±13.6	59.9±6.5	53.9±3.5	53.6±3.4	44.2±1.9	52.3±4.4	69.8±5.3	55.6±0.4
Base clusterings (best)	63.6	44.9	66.9	70.5	62.6	62.7	49.2	61.4	80.9	63.3
ECCMS [*TNNLS*, 2024]	54.2±3.1	52.6±5.9	65.3±2.5	62.0±0.7	58.1±0.7	54.9±1.8	44.1±1.2	N/A	N/A	N/A
CEAM [*TKDE*, 2024]	55.7±3.2	44.6±5.2	64.0±3.3	62.1±1.8	54.7±1.2	55.7±1.8	49.8±1.0	N/A	N/A	N/A
SPCE [*TNNLS*, 2021]	46.0±5.2	56.1±5.7	61.0±1.8	63.1±2.6	54.4±2.5	55.0±3.5	48.8±0.9	N/A	N/A	N/A
CESHL [*Inf. Fusion*, 2022]	49.2±3.3	51.4±1.0	66.2±3.1	64.5±6.2	57.4±1.8	57.0±0.5	30.8±4.2	N/A	N/A	N/A
DREC [*Neuro.*, 2019]	52.7±4.7	48.4±6.6	65.1±2.8	58.1±5.6	51.1±5.4	55.8±2.3	42.6±2.5	49.7±3.7	67.1±4.9	52.4±2.7
TRCE [*AAAI*, 2021]	52.8±2.9	**63.5±2.0**	64.5±2.2	64.2±6.7	53.2±0.6	55.5±3.0	49.8±0.9	N/A	N/A	N/A
LWEA [*TCYB*, 2018]	57.4±10.8	53.3±7.6	65.7±2.6	63.4±3.0	58.0±1.6	48.3±0.6	43.4±1.4	51.5±2.0	73.2±2.5	N/A
LWGP [*TCYB*, 2018]	57.6±5.8	54.6±5.2	65.6±2.1	61.7±0.3	55.5±2.1	48.5±1.3	48.9±1.2	51.6±3.6	71.7±3.6	57.3±2.4
ECPCS-HC [*TSMC*, 2021]	45.3±2.9	49.5±4.0	63.8±3.1	62.1±0.7	53.0±3.6	52.7±1.5	41.4±1.1	51.6±2.6	60.9±4.1	N/A
ECPCS-MC [*TSMC*, 2021]	48.6±2.9	49.9±1.8	64.2±3.0	60.1±7.1	52.6±0.4	49.7±0.7	48.6±1.1	51.2±3.7	71.8±3.2	57.8±2.1
U-SENC [*TKDE*, 2020]	55.5±2.5	48.7±1.9	66.3±2.6	62.6±7.5	53.6±1.8	53.9±2.1	51.2±1.6	51.8±3.7	73.1±3.6	58.3±0.8
SEC [*KDD*, 2015]	43.9±7.6	49.5±8.8	49.2±9.1	56.4±5.0	51.2±6.0	48.4±3.8	44.3±1.5	45.5±5.2	57.9±7.5	53.0±4.8
PTA-AL [*TKDE*, 2016]	57.8±6.3	53.7±3.5	62.3±2.2	61.4±4.4	54.4±2.0	48.9±1.1	35.5±1.5	51.5±3.6	75.3±3.3	56.5±2.8
PTA-CL [*TKDE*, 2016]	56.8±5.0	47.1±7.1	66.2±5.5	62.1±4.0	56.2±2.9	49.5±0.8	34.5±1.5	52.3±3.2	76.3±3.8	56.8±4.3
PTGP [*TKDE*, 2016]	57.3±5.1	52.1±5.0	67.0±5.5	63.8±4.8	55.4±2.0	49.4±0.8	34.5±1.2	52.0±2.5	75.7±3.8	55.9±3.5
AWMVC [*AAAI*, 2023]	54.9±4.5	47.0±6.6	51.2±5.9	64.0±4.3	51.1±1.8	52.9±1.7	50.3±0.8	50.0±4.0	72.2±4.3	56.9±2.8
SMVAGC-SF [*TIP*, 2024]	55.7±4.2	47.7±6.2	51.6±9.0	64.1±6.0	51.8±1.8	53.7±3.1	49.9±1.2	51.8±2.6	73.2±4.4	57.1±4.3
YACHT	**62.5±1.3**	58.0±8.9	**71.6±6.5**	**68.3±1.9**	**58.7±1.0**	**57.5±1.1**	**53.8±1.6**	**55.3±3.0**	**76.8±2.5**	**60.9±4.1**
					ARI					
Base clusterings (average)	21.8±8.4	3.8±2.2	34.4±11.1	47.1±4.3	36.8±2.3	36.6±2.3	39.3±1.8	36.2±2.0	57.8±3.7	38.0±2.9
Base clusterings (best)	44.1	9.2	56.7	54.5	43.1	43.3	43.7	41.1	65.7	42.2
ECCMS [*TNNLS*, 2024]	27.2±3.2	10.6±9.8	53.8±2.5	49.3±1.1	39.7±0.9	39.5±1.4	37.6±0.7	N/A	N/A	N/A
CEAM [*TKDE*, 2024]	32.5±4.5	6.2±7.6	48.4±4.9	48.9±0.9	35.9±3.5	36.6±2.1	46.0±0.7	N/A	N/A	N/A
SPCE [*TNNLS*, 2021]	27.2±5.3	9.6±7.6	46.9±4.7	50.8±1.8	38.1±0.7	38.9±1.0	47.6±1.3	N/A	N/A	N/A
CESHL [*Inf. Fusion*, 2022]	19.2±3.5	13.0±2.7	52.5±4.5	50.1±4.1	39.0±0.9	38.9±0.7	13.9±7.5	N/A	N/A	N/A
DREC [*Neuro.*, 2019]	26.1±4.5	6.6±7.4	52.4±2.1	45.3±5.4	34.6±3.8	38.1±1.4	34.1±5.2	34.7±2.2	54.5±4.5	34.5±2.3
TRCE [*AAAI*, 2021]	22.8±4.7	18.6±6.9	52.0±2.0	49.7±4.9	36.8±0.5	37.5±1.7	41.5±1.3	N/A	N/A	N/A
LWEA [*TCYB*, 2018]	35.8±15.1	12.1±8.0	52.9±3.2	50.1±2.1	39.4±0.6	39.3±0.8	37.9±0.7	37.5±1.2	59.5±1.7	N/A
LWGP [*TCYB*, 2018]	31.3±12.5	13.5±1.0	54.4±1.8	49.7±0.8	38.4±1.2	37.2±1.6	42.1±1.1	36.7±1.5	58.8±2.4	40.2±2.0
ECPCS-HC [*TSMC*, 2021]	14.6±1.8	9.0±7.6	51.9±2.4	51.2±0.8	38.1±1.7	39.4±1.1	35.3±0.8	34.5±2.8	52.4±2.6	N/A
ECPCS-MC [*TSMC*, 2021]	18.9±3.0	12.5±2.4	53.1±2.0	47.0±4.5	36.4±0.3	37.2±1.3	42.1±0.9	36.3±1.8	58.7±2.0	40.6±1.9
U-SENC [*TKDE*, 2020]	30.2±4.0	12.0±2.1	51.6±2.1	48.9±4.9	36.9±0.9	37.1±1.1	46.0±1.0	36.8±2.0	59.8±2.6	41.1±0.7
SEC [*KDD*, 2015]	15.2±6.9	2.5±15.5	30.2±13.0	41.3±5.7	33.1±5.8	30.1±4.3	35.0±4.0	29.1±6.5	43.0±8.8	34.6±4.5
PTA-AL [*TKDE*, 2016]	30.2±7.0	16.4±4.1	49.3±3.3	50.8±2.1	37.6±1.0	37.8±1.6	31.8±1.6	36.5±2.0	61.2±2.4	38.0±3.1
PTA-CL [*TKDE*, 2016]	30.1±6.2	12.0±5.1	52.0±4.3	51.7±1.4	38.7±1.4	38.5±1.3	31.1±1.4	37.6±1.7	61.9±2.8	39.1±2.7
PTGP [*TKDE*, 2016]	29.9±6.4	14.3±3.7	54.6±3.6	51.9±1.9	38.2±1.0	38.6±1.6	31.8±0.9	37.1±1.5	61.6±2.8	38.7±3.1
AWMVC [*AAAI*, 2023]	32.5±3.2	8.2±7.2	31.7±7.8	50.3±5.6	33.0±2.0	35.1±2.0	44.7±1.1	33.9±1.8	57.5±3.5	38.5±2.9
SMVAGC-SF [*TIP*, 2024]	33.0±4.6	10.3±6.4	33.0±9.7	50.3±3.9	33.7±1.5	35.6±2.5	44.1±1.0	34.4±2.8	58.1±3.9	39.6±3.4
YACHT	**41.1±4.3**	**23.1±13.5**	**54.7±5.8**	**52.6±1.8**	**41.5±0.6**	**40.9±1.1**	**49.1±0.9**	**38.4±2.3**	**62.0±3.2**	**43.0±3.0**

Table 2: Clustering performance is measured by two metrics (%). The optimal results are highlighted in bold.

Following many ensemble methods, we set the ensemble size to 20 and conduct 10 repeated experiments with different base clustering combinations, reporting the mean and standard deviation. All compared methods use the same combination of base clustering. In YACHT, we set the hyperedge weight parameter $\theta = 0.4$ and the maximum order of hyperedge random walks $d = 20$. We set the hyperparameters as follows: the number of anchors $k = k' + c$, where $k' \in \{-1, 0, 1, 2, 3\}$; and the similarity threshold $\alpha \in [0.9 : 0.01 : 1]$. All methods are evaluated using two popular clustering metrics: Accuracy (ACC), and Adjusted Rand Index (ARI). We use 17 representative methods, including state-of-the-art (SOTA) ensemble methods, fast ensemble methods and anchor-based clustering methods. • SOTA ensemble methods: ECCMS [Jia *et al.*, 2024], CEAM [Zhou *et al.*, 2024], SPCE [Zhou *et al.*, 2021b], CESHL [Zhou *et al.*, 2022], TRCE [Zhou *et al.*, 2021c],



(a) FashionMNIST (b) EMNIST

Figure 3: ACC w.r.t. k' and α.

DREC [Zhou *et al.*, 2019], ECPCS-HC/MC [Huang *et al.*, 2021]. • Fast ensemble methods: LWEA/LWGP [Huang *et al.*, 2018], U-SENC [Huang *et al.*, 2020], SEC [Liu *et al.*, 2015], PTA-AL/CL [Huang *et al.*, 2016], PTGP [Huang *et al.*, 2016]. • Anchor-based clustering methods: AWMVC [Wan *et al.*, 2023], SMVAGC-SF [Wang *et al.*, 2024].

4.2 Experimental Results

Table 2 presents the experimental results on two metrics. 'N/A' indicates excessive runtime or insufficient memory. From the experimental results, we have the following observations: (i) YACHT outperforms the average results of base clustering on all metrics across all datasets, demonstrating the effectiveness of integrating multiple base clustering results. (ii) YACHT demonstrates strong performance advantages across various datasets of different sizes, highlighting the method's excellent scalability. Particularly in terms of the ARI metric, YACHT outperforms all comparison methods across all datasets. (iii) Overall, fast ensemble methods generally do not outperform SOTA ensemble methods on most datasets. However, SOTA methods are often limited by the size of the dataset and may fail to produce results. (iv) YACHT outperforms the hypergraph method CESHL across all metrics, validating the effectiveness of using anchors to reconstruct the hypergraph.

4.3 Parameter Study and Time Comparison

Figure 3 shows the impact of different parameter combinations on the accuracy of the YACHT method. We observe that, on large-scale datasets, the YACHT method is not highly sensitive to parameter combinations, indicating that YACHT exhibits good robustness. For large-scale datasets (FashionMNIST, EMNIST), we select no more than twenty anchor points to reconstruct the hypergraph, achieving the highest clustering accuracy, which validates the effectiveness of the reconstruction. Additionally, the parameter α has some impact on the accuracy of the YACHT method, indicating that although $\mathbf{H}$ is incorporated into the optimization objective, the initial values still affect the clustering accuracy. This also suggests that the hypergraph enhancement strategy can adjust the hypergraph structure through a threshold, providing a better initial value for the optimization method. Figure 4 shows a comparison of the running time across different methods. Although the YACHT method is not the fastest ensemble clustering method, it is the first to introduce a dynamic anchor learning strategy compared to existing fixed-anchor ensemble methods. Dynamic anchor learning significantly improves

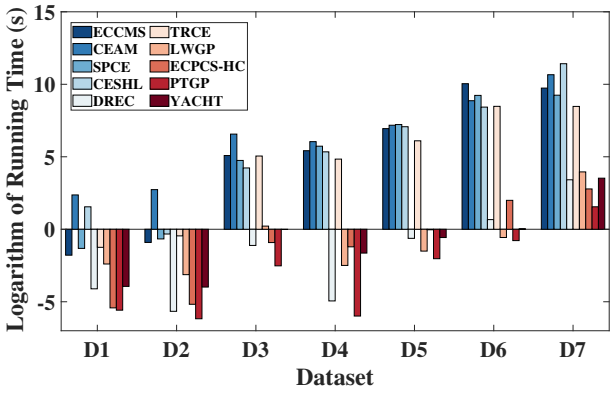


Figure 4: The running time comparison of different methods.

Dataset	Metric	w/o $\mathbf{H}$-Re	w/o $\mathbf{H}$-r	w/o $\mathbf{H}$-e	YACHT
D3	ACC	0.616	0.700	0.681	**0.716**
	ARI	0.468	0.538	0.506	**0.547**
D9	ACC	0.719	0.755	**0.768**	**0.768**
	ARI	0.586	0.610	**0.620**	**0.620**

Table 3: Ablation study results.

accuracy, but updating the anchors inevitably requires more time. In practical applications, accepting a bit of additional runtime is reasonable in order to achieve higher accuracy.

4.4 Ablation Study

Table 3 shows the results of the ablation versions of the YACHT method. 'w/o $\mathbf{H}$-Re' indicates the version without hypergraph reconstruction, where k-means is applied directly on the hypergraph. 'w/o $\mathbf{H}$-r' indicates the version using hypergraph reconstruction only, without hypergraph update and regularization. 'w/o $\mathbf{H}$-e' refers to the version without the hypergraph enhancement strategy. We observe that the clustering accuracy significantly decreases without hypergraph reconstruction, which suggests that reconstruction helps remove erroneous information, improving clustering accuracy. Overall, the absence of the hypergraph enhancement strategy and the lack of hypergraph regularizer both lead to a decrease in clustering accuracy. This demonstrates the effectiveness of hypergraph reconstruction and the hypergraph regularizer in the YACHT method.

5 Conclusion

This paper proposes a dynamic anchor-based learning and hypergraph reconstruction ensemble clustering method, YACHT. YACHT transforms the base clustering results into a hypergraph and designs a hypergraph enhancement strategy to improve its reliability. Unlike other methods, YACHT learns a high-quality anchor similarity matrix through hypergraph reconstruction. Additionally, it constructs a more reliable hypergraph and constrains its update using a local consensus alignment term and hypergraph regularization. Experiments on real datasets validate the effectiveness of YACHT. Although YACHT is not the fastest ensemble method, its accuracy significantly outperforms existing SOTA methods.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (62472294) and Grant 2024ZD0607700.

References

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

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

[Bazaraa *et al.*, 2013] Mokhtar S Bazaraa, Hanif D Sherali, and CM Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, 2013.

[Chen *et al.*, 2023] Man-Sheng Chen, Jia-Qi Lin, Chang-Dong Wang, Wu-Dong Xi, and Dong Huang. On regularizing multiple clusterings for ensemble clustering by graph tensor learning. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 3069–3077, 2023.

[Feng *et al.*, 2024] Wenyi Feng, Zhe Wang, Ting Xiao, and Mengping Yang. Adaptive weighted dictionary representation using anchor graph for subspace clustering. *Pattern Recognition*, 151:110350, 2024.

[Fern and Brodley, 2004] Xiaoli Zhang Fern and Carla E Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the 21st International Conference on Machine Learning*, page 36, 2004.

[Gao *et al.*, 2022] Yue Gao, Zizhao Zhang, Haojie Lin, Xibin Zhao, Shaoyi Du, and Changqing Zou. Hypergraph learning: Methods and practices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2548–2566, 2022.

[Hao *et al.*, 2024] Zhezheng Hao, Zhoumin Lu, Guoxu Li, Feiping Nie, Rong Wang, and Xuelong Li. Ensemble clustering with attentional representation. *IEEE Transactions on Knowledge and Data Engineering*, 36(2):581–593, 2024.

[Huang *et al.*, 2016] Dong Huang, Jian-Huang Lai, and Chang-Dong Wang. Robust ensemble clustering using probability trajectories. *IEEE Transactions on Knowledge and Data Engineering*, 28(5):1312–1326, 2016.

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

[Huang *et al.*, 2020] Dong Huang, Chang-Dong Wang, Jian-Sheng Wu, Jian-Huang Lai, and Chee-Keong Kwoh. Ultra-scalable spectral clustering and ensemble clustering. *IEEE Transactions on Knowledge and Data Engineering*, 32(6):1212–1226, 2020.

[Huang *et al.*, 2021] Dong Huang, Chang-Dong Wang, Hongxing Peng, Jianhuang Lai, and Chee-Keong Kwoh. Enhanced ensemble clustering via fast propagation of cluster-wise similarities. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(1):508–520, 2021.

[Huang *et al.*, 2023] Qirui Huang, Rui Gao, and Hoda Akhavan. An ensemble hierarchical clustering algorithm based on merits at cluster and partition levels. *Pattern Recognition*, 136:109255, 2023.

[Jia *et al.*, 2021] Yuheng Jia, Hui Liu, Junhui Hou, and Qingfu Zhang. Clustering ensemble meets low-rank tensor approximation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7970–7978, 2021.

[Jia *et al.*, 2024] Yuheng Jia, Sirui Tao, Ran Wang, and Yongheng Wang. Ensemble clustering via co-association matrix self-enhancement. *IEEE Transactions on Neural Networks and Learning Systems*, 35(8):11168–11179, 2024.

[Li *et al.*, 2021] Feijiang Li, Yuhua Qian, and Jieting Wang. Got: A growing tree model for clustering ensemble. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8349–8356, 2021.

[Li *et al.*, 2023] Hongmin Li, Xiucai Ye, Akira Imakura, and Tetsuya Sakurai. Lsec: Large-scale spectral ensemble clustering. *Intelligent Data Analysis*, 27(1):59–77, 2023.

[Li *et al.*, 2024] Taiyong Li, Xiaoyang Shu, Jiang Wu, Qingxiao Zheng, Xi Lv, and Jiaxuan Xu. Adaptive weighted ensemble clustering via kernel learning and local information preservation. *Knowledge-Based Systems*, 294:111793, 2024.

[Liang *et al.*, 2020] Yinian Liang, Zhigang Ren, Zongze Wu, Deyu Zeng, and Jianzhong Li. Scalable spectral ensemble clustering via building representative co-association matrix. *Neurocomputing*, 390:158–167, 2020.

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

[Liu *et al.*, 2024] Suyuan Liu, Ke Liang, Zhibin Dong, Siwei Wang, Xihong Yang, Sihang Zhou, En Zhu, and Xinwang Liu. Learn from view correlation: An anchor enhancement strategy for multi-view clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26151–26161, 2024.

[Shi *et al.*, 2021] Yifan Shi, Zhiwen Yu, Wenming Cao, C. L. Philip Chen, Hau-San Wong, and Guoqiang Han. Fast and effective active clustering ensemble based on density peak. *IEEE Transactions on Neural Networks and Learning Systems*, 32(8):3593–3607, 2021.

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

[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 (TKDD)*, 13(1):1–25, 2019.

[Tao *et al.*, 2021] Zhiqiang Tao, Jun Li, Huazhu Fu, Yu Kong, and Yun Fu. From ensemble clustering to subspace clustering: Cluster structure encoding. *IEEE Transactions on Neural Networks and Learning Systems*, 34(5):2670–2681, 2021.

[Wan *et al.*, 2023] Xinhang Wan, Xinwang Liu, Jiyuan Liu, Siwei Wang, Yi Wen, Weixuan Liang, En Zhu, Zhe Liu, and Lu Zhou. Auto-weighted multi-view clustering for large-scale data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 10078–10086, 2023.

[Wang *et al.*, 2024] Siwei Wang, Xinwang Liu, Suyuan Liu, Wenxuan Tu, and En Zhu. Scalable and structural multi-view graph clustering with adaptive anchor fusion. *IEEE Transactions on Image Processing*, 33:4627–4639, 2024.

[Xie *et al.*, 2024] Fangyuan Xie, Feiping Nie, Weizhong Yu, and Xuelong Li. Parameter-free ensemble clustering with dynamic weighting mechanism. *Pattern Recognition*, 151:110389, 2024.

[Xu *et al.*, 2024] Jiaxuan Xu, Taiyong Li, and Lei Duan. Enhancing ensemble clustering with adaptive high-order topological weights. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 16184–16192, 2024.

[Yang *et al.*, 2023] Geping Yang, Sucheng Deng, Can Chen, Yiyang Yang, Zhiguo Gong, Xiang Chen, and Zhifeng Hao. Litewsec: a lightweight framework for web-scale spectral ensemble clustering. *IEEE Transactions on Knowledge and Data Engineering*, 35(10):10035–10047, 2023.

[Zhang *et al.*, 2018] Zizhao Zhang, Haojie Lin, and Yue Gao. Dynamic hypergraph structure learning. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 3162–3169, 2018.

[Zhang *et al.*, 2024a] Runxin Zhang, Shuaijun Hang, Zhensheng Sun, Feiping Nie, Rong Wang, and Xuelong Li. Anchor-based fast spectral ensemble clustering. *Information Fusion*, page 102587, 2024.

[Zhang *et al.*, 2024b] Xu Zhang, Yuheng Jia, Mofei Song, and Ran Wang. Similarity and dissimilarity guided co-association matrix construction for ensemble clustering. *arXiv preprint arXiv:2411.00904*, 2024.

[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.

[Zhong and Pun, 2022] Guo Zhong and Chi-Man Pun. Improved normalized cut for multi-view clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):10244–10251, 2022.

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

[Zhou *et al.*, 2021a] Peng Zhou, Liang Du, and Xuejun Li. Self-paced consensus clustering with bipartite graph. In *Proceedings of the 29th International Conference on International Joint Conferences on Artificial Intelligence*, pages 2133–2139, 2021.

[Zhou *et al.*, 2021b] Peng Zhou, Liang Du, Xinwang Liu, Yi-Dong Shen, Mingyu Fan, and Xuejun Li. Self-paced clustering ensemble. *IEEE Transactions on Neural Networks and Learning Systems*, 32(4):1497–1511, 2021.

[Zhou *et al.*, 2021c] Peng Zhou, Liang Du, Yi-Dong Shen, and Xuejun Li. Tri-level robust clustering ensemble with multiple graph learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11125–11133, 2021.

[Zhou *et al.*, 2022] Peng Zhou, Xia Wang, Liang Du, and Xuejun Li. Clustering ensemble via structured hypergraph learning. *Information Fusion*, 78:171–179, 2022.

[Zhou *et al.*, 2024] Peng Zhou, Boao Hu, Dengcheng Yan, and Liang Du. Clustering ensemble via diffusion on adaptive multiplex. *IEEE Transactions on Knowledge and Data Engineering*, 36(4):1463–1474, 2024.

[Zhuge *et al.*, 2019] Wenzhang Zhuge, Chenping Hou, Xinwang Liu, Hong Tao, and Dongyun Yi. Simultaneous representation learning and clustering for incomplete multi-view data. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 4482–4488, 2019.