A Simple yet Effective Hypergraph Clustering Network

Qianqian Wang[1], **Bowen Zhao**[1*], **Zhengming Ding**[2], **Xiangdong Zhang**[1] and **Quanxue Gao**[1]
[1]School of Telecommunications Engineering, Xidian University, Xi'an, China
[2]Department of Computer Science, Tulane University, New Orleans, LA
qqwang@xidian.edu.cn, bw_zhao@stu.xidian.edu.cn, zding1@tulane.edu, xdchen@mail.xidian.edu.cn, qxgao@xidian.edu.cn

Abstract

Hypergraph Clustering has gained significant attention due to its capability of capturing high-order structural information. Among different approaches, contrastive learning-based methods leverage self-supervised learning and data augmentation, exhibiting impressive performance. However, most of them come with the following limitations: 1) Augmentation strategies like feature dropout can potentially disrupt the intrinsic clustering structure of hypergraphs. 2) High computational demands hinder their real-world application. To address the above issues, we propose a simple yet effective **H**ypergraph **C**lustering **N**etwork framework (HCN). Specifically, HCN replaces the hypergraph convolution operation with smoothing preprocessing, which avoids high computational complexity. Besides, to retain intrinsic structure, it develops two key modules: the self-diagonal consistency module and the structure alignment module. They respectively align the similarity matrix with the identity matrix and the structural affinity matrix, which ensures intra-cluster compactness and inter-cluster separability. Extensive experiments on five benchmark datasets demonstrate HCN's superiority over state-of-the-art methods.

1 Introduction

Hypergraphs have emerged as a compelling extension of traditional graphs that can effectively capture high-order relationships among entities [Bretto, 2013; Antelmi *et al.*, 2023; Kim *et al.*, 2024]. Unlike traditional graphs, which model pairwise relationships through edges, hypergraphs extend this concept by introducing hyperedges that simultaneously connect multiple nodes. This unique property makes hypergraphs highly effective in modeling complex data structures, such as social networks [Li *et al.*, 2013; Zhu *et al.*, 2018], biological systems [Feng *et al.*, 2021; Dai and Gao, 2023], and recommendation systems [Xia *et al.*, 2021; Yu *et al.*, 2021], where interactions often go beyond simple pairwise relationships.

*Corresponding Author

Inspired by the remarkable success of graph neural networks (GNNs) [Chen *et al.*, 2024b; Chen *et al.*, 2024a; Chen *et al.*, 2025], recent research has increasingly focused on the development of hypergraph neural networks (HNNs) [Feng *et al.*, 2019; Cai *et al.*, 2022; Wang *et al.*, 2023], aiming to leverage hypergraphs to characterize complex relationships. HNNs iteratively aggregate node features into their incident hyperedges and then propagate the hyperedge features back to their incident nodes. Although these methods demonstrate promising results, they primarily focus on (semi-)supervised node classification tasks. However, in real-world scenarios, there exist massive unlabeled data due to the high expense and complexity of hypergraph data annotation. These methods depend on label guidance and confront difficulties when processing the unlabeled data. Therefore, several unsupervised hypergraph learning methods were developed to mitigate this problem, among which, hypergraph clustering is the most representative approach.

Hypergraph clustering aims to partition the nodes of a hypergraph into several disjoint groups in an unsupervised manner. Existing hypergraph clustering methods can be broadly categorized into traditional and deep learning-based methods. For instance, traditional hypergraph clustering methods include applying spectral clustering to hypergraphs [Zhou *et al.*, 2006; Yang *et al.*, 2021]. Nevertheless, these methods are inadequate for capturing complex and high-order structural information, while deep approaches can effectively mitigate this issue. Especially, contrastive learning-based hypergraph clustering [Wei *et al.*, 2022; Xia *et al.*, 2022; Wu *et al.*, 2024; Qian *et al.*, 2024] significantly improves representation capabilities as a self-supervised method. Most of them typically employ various data augmentation strategies, such as simple feature dropout or structural perturbation, to generate distinct views of the hypergraph. Subsequently, they utilize HNNs to learn representations for each view and employ contrastive learning to maximize the similarity between positive pairs while minimizing the similarity between negative pairs.

Although hypergraph contrastive clustering has achieved promising results, they have the following drawbacks: 1) The data augmentation strategies can inadvertently distort the semantic information of the hypergraph, which risks altering the intrinsic clustering structure of the hypergraph and undermining its ability to accurately capture high-order dependencies. 2) HNNs typically require iterative updates of node

representations during training process, which results in high computational complexity. Therefore, a simple yet effective hypergraph clustering network is highly expected.

In this paper, we propose a Hypergraph Clustering Network (HCN) to address the aforementioned issues. Specifically, it first implements a hypergraph smoothing preprocessing step instead of the hypergraph convolution operation, effectively reducing the computational complexity. Then, we feed the preprocessed hypergraph data into unshared Multilayer Perceptron (MLP) encoders to generate distinct views. Additionally, HCN incorporates a self-diagonal consistency module to enhance consistency by aligning the similarity matrices across different views with the identity matrix, thereby preserving shared semantic information. In parallel, it introduces the structure alignment module to capture the intrinsic structural patterns by aligning the similarity matrices with the hypergraph's structural affinity matrix, which ensures intra-cluster compactness and inter-cluster separability. Finally, we fuse the representations from different views for clustering. The major contributions are summarized as follows:

- We propose a simple yet effective hypergraph clustering network, which explores hypergraph learning in clustering tasks.
- We utilize hypergraph smoothing preprocessing instead of hypergraph convolution for structural information representation with better computational efficiency.
- We introduce the self-diagonal consistency module and structure alignment module to maintain consistency and simultaneously capture the intrinsic hypergraph structural patterns.

2 Related Work

In recent years, advancements in HNNs have demonstrated significantly improved performance by effectively exploring high-order information inherent in complex data structures. HGNN [Feng *et al.*, 2019] is the first work to propose hypergraph neural networks, extending GCNs to a more general form. Unignn [Huang and Yang, 2021] is a unified framework that extends GNNs to hypergraphs, enabling seamless adaptation of advanced GNN architectures for effective hypergraph representation learning. HSL [Cai *et al.*, 2022] refines hypergraph structures and trains HNNs end-to-end using sampling and contrastive learning to enhance performance. AllSet [Chien *et al.*, 2021] introduces a highly flexible and expressive hypergraph neural network framework by leveraging multiset functions. However, these methods primarily focus on node classification guided by label information, making them unsuitable for unsupervised scenarios.

Given the powerful representation learning capabilities of contrastive learning, researchers have leveraged it for hypergraph representation learning. HyperGCL [Wei *et al.*, 2022] improves hypergraph neural networks in low-label settings by combining domain-driven and generative augmentation strategies to construct effective contrastive views. MMACL [Lee and Chae, 2024] integrates high-order and pairwise relationships through mixed-attention mechanisms and leverages multi-view contrastive learning for more expressive node representations. TriCL [Lee and Shin, 2023] introduces tri-directional contrastive learning on hypergraphs, capturing both node- and group-level structural information through three complementary contrastive objectives. CHGNN [Song *et al.*, 2024] combines adaptive view generation, enhanced hypergraph encoding, and contrastive learning to achieve superior classification accuracy on hypergraphs. CCL [Wu *et al.*, 2024] alternates contrastive learning between standard graphs and hypergraphs to collaboratively enhance node representations. However, on the one hand, inappropriate data augmentation strategies may alter the semantic information of hypergraphs, thereby disrupting the intrinsic clustering structure of hypergraphs. On the other hand, these approaches often involve substantial computational overhead, which hinders their practicality in real-world scenarios.

3 The Proposed Method

In this section, we introduce a novel framework for a Hypergraph Clustering Network (HCN). As illustrated in Figure 1, the framework is built upon three pivotal modules: hypergraph data preprocessing, self-diagonal consistency module, and structure alignment module. The subsequent sections will provide an in-depth exploration of each module, outlining its mechanisms and contributions to the overall approach.

3.1 Notations and Problem Definition

Notations: Consider a hypergraph data $\mathcal{H} = \{\mathbf{X}, \mathbf{H}, \mathcal{E}\}$, let $\mathcal{V} = \{v_1, v_2, ..., v_n\}$ represents a set of n nodes belonging to k distinct classes, and $\mathcal{E} = \{e_1, e_2, ..., e_m\}$ denotes a set of hyperedges. The attribute matrix of all nodes is given by $\mathbf{X} \in \mathbb{R}^{n \times d}$, where d indicates the dimension of the node attributes. The incidence matrix $\mathbf{H} \in \{0, 1\}^{n \times m}$ encodes the relationship between nodes and hyperedges, where $\mathbf{H}_{ij} = 1$ if node v_i is contained in hyperedge e_j, and $\mathbf{H}_{ij} = 0$ otherwise. We define $\mathbf{D}_v \in \mathbb{R}^{n \times n}$ as the diagonal degree matrix of nodes and $\mathbf{D}_\mathcal{E} \in \mathbb{R}^{m \times m}$ as the diagonal degree matrix of hyperedges. For node v_i and hyperedge e_k, $\mathbf{D}_v^i$ represents the number of hyperedges connected to v_i, while $\mathbf{D}_\mathcal{E}^k$ denotes the number of nodes contained in e_k.

Problem Statement: The objective of deep hypergraph clustering is to encode nodes using a neural network in an unsupervised manner and then partition them into several disjoint groups. In general, the hypergraph neural network $\mathcal{F}$ embeds the node attribute $\mathbf{X}$ and structural information $\mathbf{H}$ of the hypergraph into low-dimensional representations $\mathbf{Z}$. Subsequently, clustering algorithms such as k-means or spectral clustering are employed to partition these low-dimensional representations into k disjoint groups.

3.2 Hypergraph Data Preprocessing

Existing hypergraph methods employ hypergraph neural networks to capture meaningful node features by iteratively updating hyperedge representations via their incident nodes and node representations via their incident hyperedges [Feng *et al.*, 2019; Gao *et al.*, 2022; Lee and Shin, 2023; Kim *et al.*, 2024]. However, these methods often require substantial computational resources, which restricts their practical applicability.

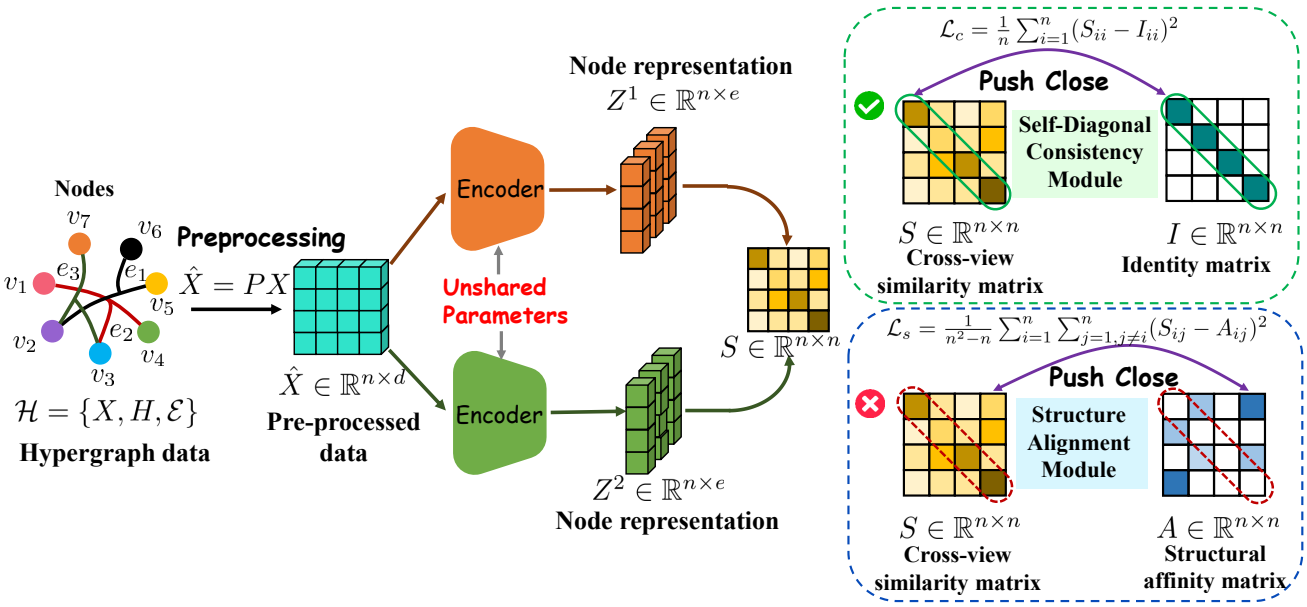


Figure 1: The framework of our proposed HCN. HCN consists of three key modules: hypergraph data preprocessing (HDP), self-diagonal consistency module (SDC), and structure alignment module (SAM). Specifically, HDP separates the exploitation of hypergraph structural information from the model training process. SDC ensures the consistency of individual node representations across different views by aligning cross-view similarity matrices with the identity matrix, thereby preserving shared semantic information while mitigating view-specific noise. SAM aligns node representations with the hypergraph's structural relationships by minimizing the discrepancy between the off-diagonal elements of the similarity matrix and the structural affinity matrix, ensuring that nodes within the same hyperedge exhibit consistent semantic properties.

Inspired by [Tang *et al.*, 2024], we utilize hypergraph smoothing preprocessing instead of hypergraph convolution for structural information representation, thereby reducing the computational burden. Specifically, we first define the hyperedge weight matrix, which is calculated as:

$$\mathbf{M} = \sum_{k=1}^{m} \frac{\mathbf{H}_{ik} \cdot \mathbf{H}_{jk}}{\mathbf{D}_{\mathcal{E}}^{k}}. \tag{1}$$

That is, the numerator equals 1 if e_k contains both v_i and v_j simultaneously; otherwise, it equals 0. Then, we generate the hypergraph propagation matrix $\mathbf{P}$ as follows:

$$\begin{aligned}\mathbf{P} = &(1-\alpha)^t(\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{M}}\hat{\mathbf{D}}^{-1/2})^t \\ &+ \alpha\sum_{l=0}^{t-1}(1-\alpha)^l(\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{M}}\hat{\mathbf{D}}^{-1/2})^l,\end{aligned} \tag{2}$$

where $\hat{\mathbf{M}} = \mathbf{M} + \mathbf{I}_n$, and $\hat{\mathbf{D}} \in \mathbb{R}^{n \times n}$ denotes the diagonal degree matrix corresponding to $\hat{\mathbf{M}}$. t represents the number of propagation layers. Specifically, each propagation step extends the spread of node information to more distant neighbors. When $t = 1$, the propagation is limited to direct neighbors, while $t > 1$ indicates that the information is propagated to nodes at greater distances. Additionally, a high value of α places greater emphasis on preserving the intrinsic information of each node, whereas a lower α amplifies the contribution of information from neighboring nodes. Then, we obtain the preprocessed hypergraph data, which can be represented as follows:

$$\hat{\mathbf{X}} = \mathbf{P}\mathbf{X}, \tag{3}$$

where $\hat{\mathbf{X}}$ denotes the preprocessed hypergraph data. Through the preprocessing process, we can train the processed hypergraph data using a simple network, significantly reducing the training time.

3.3 Self-Diagonal Consistency Module

The preprocessed hypergraph data is input into MLP encoders to generate node representations, which can be formulated as follows:

$$\mathbf{Z}^v = \mathrm{MLP}_v(\hat{\mathbf{X}}), \quad v = \{1, 2\}, \tag{4}$$

where MLP_1 and MLP_2 represent two unshared MLP encoders, while $\mathbf{Z}^1$ and $\mathbf{Z}^2$ correspond to the first and second views of node representations, respectively. By mapping $\hat{\mathbf{X}}$ into different subspaces, this approach captures diverse semantic information. Unlike other data augmentation techniques (e.g., feature dropout), this method preserves the hypergraph's structural and attribute integrity, thereby avoiding alterations to its semantic information.

To ensure consistency among node representations across different views, we introduce the self-diagonal consistency module. Specifically, the similarity between the two node representations, $\mathbf{Z}^1$ and $\mathbf{Z}^2$, is formally defined as follows:

$$\mathbf{S} = \frac{\mathbf{Z}^1 \cdot (\mathbf{Z}^2)^\top}{\|\mathbf{Z}^1\| \, \|\mathbf{Z}^2\|}, \tag{5}$$

where $\mathbf{S} \in \mathbb{R}^{n \times n}$ is a cross-view similarity matrix. To keep the consistency of individual nodes across different views, we align the similarity matrix with the identity matrix. This can be expressed as:

$$\mathcal{L}_c = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{S}_{ii} - \mathbf{I}_{ii})^2, \tag{6}$$

where $\mathbf{I} \in \mathbb{R}^{n \times n}$ is an identity matrix. By minimizing this loss, the alignment of corresponding node representations across views is effectively promoted. This ensures that each node's representation remains consistent and invariant across different views, preserving the shared semantic information while mitigating the influence of view-specific noise.

3.4 Structure Alignment Module

Considering the structural information inherent in hypergraphs, nodes within the same hyperedge are more likely to exhibit similar semantic characteristics. To effectively capture and reinforce this property, we propose the structure alignment module. This module aims to align the relationships between nodes within the hypergraph structure, ensuring the consistency of their representations. Specifically, to quantify the relationships between nodes, we compute the product of the incidence matrix $\mathbf{H}$ and its transpose. Formally, it is defined as follows:

$$\mathbf{A} = \mathbf{H}\mathbf{H}^\top, \tag{7}$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$ represents the structural affinity matrix. Here, $\mathbf{A}_{ij}$ encodes the strength of the relationship between nodes v_i and v_j based on their shared membership in hyperedges.

To ensure structural consistency, we align the off-diagonal elements of the cross-view similarity matrix $\mathbf{S}$ with the corresponding elements of $\mathbf{A}$. The structure alignment loss is thus defined as:

$$\mathcal{L}_s = \frac{1}{n^2 - n} \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} (\mathbf{S}_{ij} - \mathbf{A}_{ij})^2. \tag{8}$$

By minimizing $\mathcal{L}_s$, the structure alignment module encourages the learned representations to align with the hypergraph structure, promoting semantic coherence among nodes that share hyperedges. This ensures that the hypergraph's intrinsic connectivity patterns are preserved, ultimately enhancing the quality and reliability of the node representations across different views.

3.5 The Objective Function

The objective function of the proposed HCN framework integrates the self-diagonal consistency loss $\mathcal{L}_c$ and the structure alignment loss $\mathcal{L}_s$. In summary, the objective function is defined as follows:

$$\mathcal{L} = \mathcal{L}_c + \beta \mathcal{L}_s, \tag{9}$$

where β is a trade-off parameter that balances the contributions of $\mathcal{L}_c$ and $\mathcal{L}_s$.

To obtain the final clustering results, we first fuse node representations $\mathbf{Z}^1$ and $\mathbf{Z}^2$ as follows:

$$\mathbf{Z} = \frac{1}{2}(\mathbf{Z}^1 + \mathbf{Z}^2). \tag{10}$$

	CORA	CITESEER	PUBMED	CORA-CA	20NEWS
Nodes	1,434	1,458	3,840	2,388	16,242
Hyperedges	1,579	1,079	7,963	1,072	100
Memberships	4,786	3,453	34,629	4,585	65,451
Features	1,433	3,703	500	1,433	100
Classes	7	6	3	7	4

Table 1: Statistical characteristics of five datasets used in our experiments.

The fused representation $\mathbf{Z}$ is then utilized as input to the k-means clustering to partition the nodes into multiple disjoint groups.

4 Experiments

4.1 Experimental Settings

Benchmark Datasets. The experiments are conducted on five benchmark hypergraph datasets, including CORA [Sen *et al.*, 2008], CITESEER [Sen *et al.*, 2008], PUBMED [Sen *et al.*, 2008], CORA-CA [Rossi and Ahmed, 2015], and 20NEWS [Dua and Graff, 2017]. Among these datasets, CORA, CITESEER, and PUBMED are co-citation datasets, while CORA-CA is a co-authorship dataset. The 20NEWS dataset is sourced from the UCI Categorical Machine Learning Repository. The detailed characteristics and statistics of datasets are provided in Table 1.

Comparison Methods. To validate the effectiveness of the proposed HCN, we perform a comparative analysis against state-of-the-art baseline methods, encompassing four deep graph clustering methods and four deep hypergraph methods. Deep graph clustering methods include CCGC [Yang *et al.*, 2023a], HSAN [Liu *et al.*, 2023], CONVERT [Yang *et al.*, 2023b], NS4GC [Liu *et al.*, 2024], while deep hypergraph methods encompass HyperGCL [Wei *et al.*, 2022], MMACL [Lee and Chae, 2024], TriCL [Lee and Shin, 2023], CHGNN [Song *et al.*, 2024].

Evaluation Metrics. We assess clustering performance using four standard metrics: Accuracy (ACC), Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), and Macro F1-score (F1), where higher values indicate superior performance. To avoid the influence of randomness, the mean and standard deviation of these metrics are calculated over 10 independent runs for each method.

Implementation Details. The experiments are conducted on a system equipped with an Intel Core i9-13900K CPU, an NVIDIA GeForce RTX 4090 GPU, and 64GB of RAM. All experiments are implemented using the Pytorch framework, with a maximum training epoch limit of 400. The Adam optimizer [Kingma, 2014] is used to minimize the total loss, and the k-means algorithm is applied to the fused embeddings to obtain the final clustering results. For deep graph clustering methods, since these methods cannot be directly applied to hypergraphs, we transform the hypergraphs into graphs using clique expansion before applying them.

4.2 Comparison Results

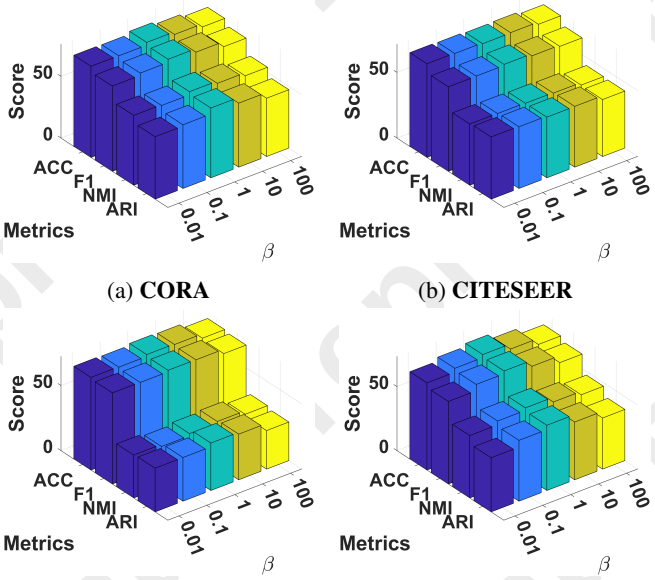
In Table 2, we comprehensively compare our proposed method with the other eight state-of-the-art methods on four

Dataset	Metric	Graph Methods				Hypergraph Methods				
		CCGC	HSAN	CONVERT	NS4GC	HyperGCL	MMACL	TriCL	CHGNN	HCN
CORA	ACC	69.75±2.58	71.95±0.73	71.46±1.26	73.11±2.12	57.29±3.78	73.50±0.61	**75.46±1.42**	52.63±3.58	**75.95±0.41**
	NMI	50.59±3.55	53.32±0.74	52.61±1.18	54.46±2.99	38.51±3.09	56.16±0.93	**57.20±0.98**	41.21±2.35	**58.52±0.59**
	ARI	44.94±4.28	47.83±0.90	46.89±1.48	48.64±3.68	31.08±3.93	49.63±0.71	**55.60±1.76**	30.15±3.41	**56.65±0.69**
	F1	64.97±2.72	67.99±1.35	67.12±1.97	69.81±1.87	55.99±3.74	69.37±2.33	**72.69±2.76**	46.33±5.04	**73.54±0.48**
CITESEER	ACC	67.97±2.30	69.72±0.62	65.84±1.02	69.47±0.92	49.11±2.54	68.78±0.78	**69.73±0.79**	50.96±2.26	**71.67±0.86**
	NMI	42.64±2.49	**44.61±0.68**	40.04±1.40	43.28±1.26	24.81±3.28	44.93±1.06	44.27±0.91	30.78±2.04	**46.85±0.63**
	ARI	41.93±3.07	45.24±1.28	40.27±1.67	44.68±1.75	22.85±3.43	45.13±0.99	**45.95±1.34**	28.36±3.03	**47.75±1.26**
	F1	61.05±2.40	63.43±1.78	60.84±1.45	61.70±2.03	45.07±2.56	62.25±1.05	**63.74±1.25**	44.02±1.66	**64.55±1.70**
PUBMED	ACC	64.84±0.78	65.96±1.00	**70.17±2.08**	68.08±0.58	60.96±5.39	68.42±0.31	68.46±1.39	62.37±3.58	**72.45±0.38**
	NMI	28.55±0.54	28.34±0.65	29.25±2.13	27.98±0.35	20.32±4.75	31.69±1.07	**33.34±0.83**	23.99±4.34	**33.07±0.79**
	ARI	25.71±0.74	26.67±1.10	**31.62±3.47**	28.97±0.57	19.94±5.65	29.87±0.48	30.43±1.44	21.57±4.53	**36.72±0.79**
	F1	63.85±0.76	65.09±1.00	**68.33±2.26**	67.37±0.61	60.35±5.74	67.97±0.30	67.82±1.34	62.32±3.73	**71.01±0.28**
CORA-CA	ACC	62.45±2.30	64.02±1.39	64.70±2.31	67.45±0.87	47.42±4.93	71.27±0.94	**72.04±1.51**	42.50±0.90	**73.59±0.74**
	NMI	42.12±1.98	44.86±0.92	42.09±2.25	46.05±2.10	28.66±4.32	51.60±1.36	**51.87±2.44**	29.44±1.02	**53.66±1.20**
	ARI	37.24±2.65	37.83±1.81	37.10±2.95	41.65±1.75	21.75±4.83	46.43±0.91	**47.96±2.07**	16.78±4.04	**51.04±1.68**
	F1	56.57±3.84	60.32±4.31	63.51±2.28	64.18±2.94	44.44±4.71	67.80±2.00	**69.28±1.41**	41.74±1.70	**71.68±0.69**
20NEWS	ACC	56.93±0.79	OOM	58.46±1.27	65.83±1.08	65.24±3.47	OOT	**71.62±0.81**	69.35±3.12	**74.85±1.14**
	NMI	20.76±1.64		22.77±1.54	31.49±1.12	30.17±3.30		**40.71±2.05**	39.22±2.48	**41.83±1.57**
	ARI	24.75±1.05		27.31±1.70	33.22±1.02	34.48±4.51		**44.04±2.71**	40.29±4.64	**49.90±1.89**
	F1	48.82±0.85		52.81±1.07	57.33±1.48	61.40±3.51		**66.66±1.99**	64.70±4.61	**70.57±1.51**

Table 2: The clustering performance is assessed across ten runs on five benchmark datasets using four evaluation metrics, with results reported as mean values accompanied by standard deviations. The best and second-best performances are highlighted in **Red** and **Blue**, respectively. 'OOM'denotes the out-of-memory failure, while 'OOT'signifies no results are obtained within 24 hours.

Dataset	Modules			Metrics			
	HDP	SDC	SAM	ACC	NMI	ARI	F1
CORA	✔	✔	✗	74.24±0.81	57.17±1.15	51.08±1.34	70.52±0.86
	✔	✗	✔	72.75±1.18	56.12±0.72	49.34±1.11	68.47±1.82
	✗	✔	✔	66.03±1.71	45.62±1.61	39.62±1.79	64.32±1.83
	✔	✔	✔	**75.95±0.41**	**58.52±0.59**	**56.65±0.69**	**73.54±0.48**
CITESEER	✔	✔	✗	71.42±0.85	46.33±0.58	47.26±1.17	63.89±1.77
	✔	✗	✔	67.01±1.92	41.90±1.92	41.19±2.58	61.98±1.56
	✗	✔	✔	67.52±1.19	41.34±0.75	41.09±1.30	61.54±1.18
	✔	✔	✔	**71.67±0.86**	**46.85±0.63**	**47.75±1.26**	**64.55±1.70**
PUBMED	✔	✔	✗	71.22±1.76	33.16±1.57	33.71±2.44	70.11±1.59
	✔	✗	✔	68.17±0.96	31.14±1.83	29.26±1.63	66.96±1.37
	✗	✔	✔	65.62±0.41	26.10±0.47	25.55±0.48	65.63±0.43
	✔	✔	✔	**72.45±0.38**	**33.07±0.79**	**36.72±0.79**	**71.01±0.28**
CORA-CA	✔	✔	✗	69.18±1.55	49.50±1.71	43.63±1.56	66.44±2.14
	✔	✗	✔	64.15±1.91	47.24±1.15	37.84±1.82	59.75±3.39
	✗	✔	✔	64.64±2.22	43.66±1.41	36.47±3.70	60.97±3.24
	✔	✔	✔	**73.39±0.74**	**53.66±1.20**	**51.04±1.68**	**71.68±0.69**
20NEWS	✔	✔	✗	69.16±0.71	40.16±1.43	42.14±1.37	65.15±1.26
	✔	✗	✔	69.47±0.52	40.78±1.67	42.47±0.49	66.63±0.21
	✗	✔	✔	67.39±0.95	36.29±3.23	37.95±1.28	63.70±2.03
	✔	✔	✔	**74.85±1.14**	**41.83±1.57**	**47.90±1.89**	**70.57±1.51**

Table 3: Ablation study on five datasets. ✔ denotes HCN with this module, ✗ denotes HCN without this module.



(a) **CORA** (b) **CITESEER** (c) **PUBMED** (d) **CORA-CA**

Figure 2: The sensitivity analysis of hyper-parameter β across four evaluation metrics on four datasets.

evaluation metrics, where the best and the sub-optimal performance are denoted in red and blue, respectively. From the results, we have the following observations:

- Our method consistently outperforms all compared approaches across most datasets, demonstrating its robustness and effectiveness. Notably, on the 20NEWS dataset, our approach achieves substantial improvements over the second-best method, TriCL, with performance gains of 3.23% in ACC, 1.12% in NMI, 5.86% in ARI, and 3.91% in F1. These significant improvements underscore the effectiveness of our model in leveraging hypergraph structure information to capture meaningful and complex relationships among nodes, leading to more precise clustering results.
- Compared with four other state-of-the-art contrastive learning-based hypergraph methods, our method shows significant advantages. These methods employ various

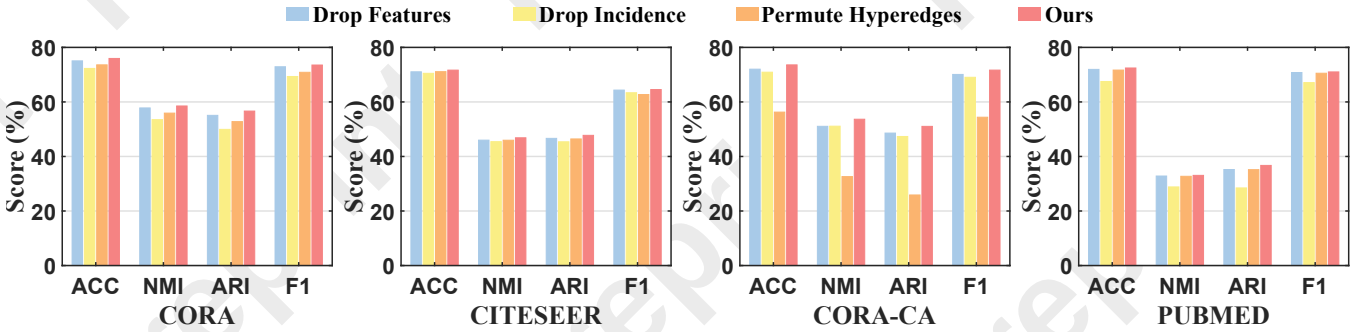


Figure 3: The experimental results of different data augmentation strategies across four datasets.

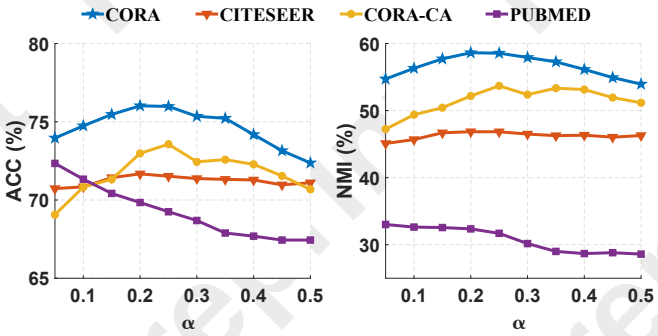


Figure 4: The sensitivity analysis of hyper-parameter α across two evaluation metrics on four datasets.

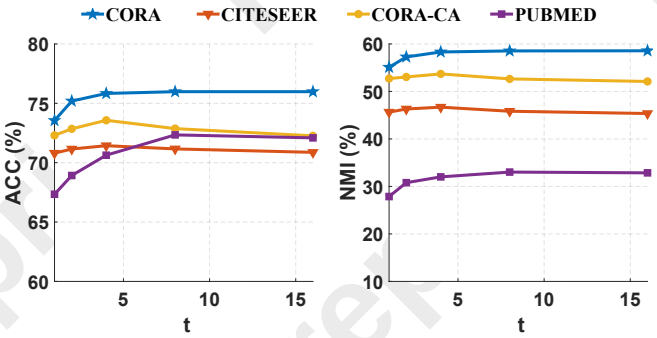


Figure 5: The sensitivity analysis of hyper-parameter t across two evaluation metrics on four datasets.

data augmentation strategies to generate different views, treating the same node across views as positive samples and different nodes as negative samples. In contrast, our method makes full use of the inherent structure of the hypergraph to define positive and negative pairs in a more effective manner. Nodes that share similar structural properties are regarded as positive samples, while those with distinct structural characteristics are treated as negative samples. Another critical distinction is that many contrastive learning-based methods rely on simple feature dropout or structural perturbation augmentations to construct different views, which can inadvertently alter the semantic information of the hypergraph. Our method, however, avoids any modification to the hypergraph itself. Instead, we employ unshared encoders to map the hypergraph into different subspaces. This approach allows for richer and more discriminative representations without the risk of information loss caused by augmentation techniques.

4.3 Ablation Studies

In this subsection, we conduct ablation experiments to verify the effectiveness of each module.

Ablation on Model Modules. We start by examining the contributions of key modules in our proposed framework. For simplicity, we denote hypergraph data preprocessing, self-diagonal consistency module, and structure alignment module as 'HDP', 'SDC', and 'SAM', respectively. From the results in Table 3, the best performance can be achieved when all modules are considered. Omitting any single module results in a noticeable decline in clustering accuracy, emphasizing the critical role each plays in enhancing the method's overall performance.

Ablation on Data Augmentation Strategies. We also conduct an in-depth analysis of the impact of different data augmentation strategies on the results. Specifically, three strategies are evaluated: drop features, drop incidence, and permute hyperedges. The detailed experimental results are illustrated in Figure 3, which clearly demonstrate that our strategies achieves superior performance compared to other strategies. Unlike the other three augmentation strategies, our method does not alter the hypergraph itself. Instead, we use unshared encoders to project the hypergraph into distinct subspaces, thereby preserving its semantic information.

4.4 Parameter Sensitivity Analysis

In this subsection, we study the sensitivity of hyper-parameters: the trade-off hyper-parameter β, the number of layers t, and hyper-parameter α on four datasets.

Effect of hyper-parameter β. To evaluate the impact of hyper-parameter β, we conduct sensitivity analysis across four benchmark datasets: CORA, CITESEER, PUBMED, and CORA-CA. The results, presented in Figure 2, illustrate the performance variations under different values of β, which are chosen from the set $\{0.01, 0.1, 1, 10, 100\}$. From the results, we observe that the clustering performance remains relatively stable with minor fluctuations as β increases. Notably, the best overall performance is achieved when $\beta = 1$, and we

Dataset	Graph Methods				Hypergraph Methods				
	CCGC	HSAN	CONVERT	NS4GC	HyperGCL	MMACL	TriCL	CHGNN	Ours
CORA	7.90	**5.07**	10.13	5.41	34.25	658.67	36.56	16.39	**3.36**
CITESEER	6.25	5.71	11.58	**4.75**	29.18	540.10	27.72	12.73	**2.56**
PUBMED	13.86	21.03	9.31	**8.39**	160.99	9792.04	218.17	734.85	**2.83**
CORA-CA	11.50	10.78	19.03	**8.47**	38.18	1048.05	34.10	17.12	**3.71**
20NEWS	52.78	OOM	53.83	429.53	181.38	OOT	139.60	**23.02**	**22.50**
Avg.	**18.46**	-	20.78	91.31	88.80	-	91.23	160.82	**6.99**

Table 4: Comparison of training time with eight baselines, with all results reported in seconds. The best and second-best results are highlighted in **Red** and **Blue**, respectively. 'Avg.' indicates the average training time across the five datasets.

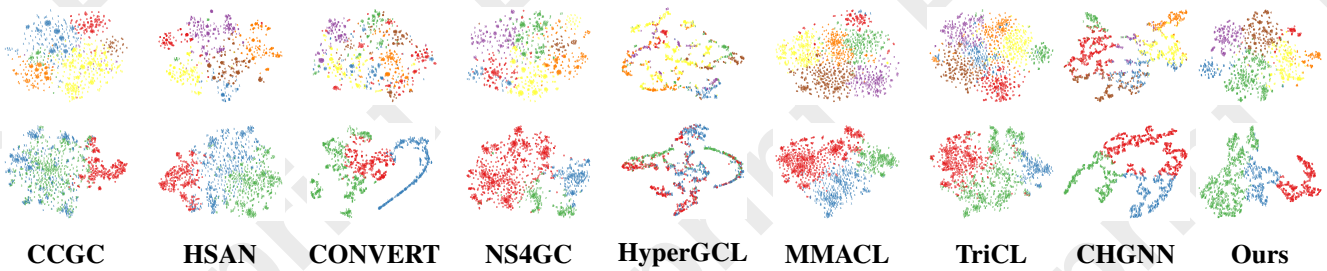


Figure 6: The visualization of consensus node representation using the t-SNE algorithm. The first row and second row correspond to CORA and PUBMED, respectively.

set $\beta = 1$ as the default value for subsequent experiments.

Effect of hyper-parameter α and t. In addition, we further investigate how the hyper-parameter α and t in the hypergraph data preprocessing module influence clustering performance. Following by [Tang *et al.*, 2024], α is selected from the set $\{0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}$, while t is chosen from $\{1, 2, 4, 8, 16\}$. Figure 4 and 5 illustrate the variations in ACC and NMI of our approach as α and t change. From Figure 4, we observe that for the PUBMED dataset, clustering performance decreases as α increases. In contrast, for the CORA and CORA-CA datasets, clustering performance improves with increasing α until it reaches a peak, after which it gradually declines. A larger α value places greater emphasis on preserving the intrinsic information of the node, whereas a smaller α value amplifies the impact of information derived from neighboring nodes. From Figure 5, we can see that as t increases, clustering performance gradually improves before leveling off. Based on these observations, selecting appropriate values of α and t is critical for achieving optimal clustering performance.

4.5 Efficiency Comparison

We evaluate the training efficiency of HCN by comparing it with eight baselines on five datasets under identical 400-epoch settings. As shown in Table 4, HCN achieves the fastest training speed across all datasets, outperforming recent contrastive hypergraph methods by at least 12× on average. This efficiency stems from two key designs: (1) Unlike the hypergraph neural networks used in hypergraph methods, our architecture consists of unshared MLPs, significantly reducing the number of network parameters. (2) The hypergraph data preprocessing module employs smoothing preprocessing instead of hypergraph convolution, significantly simplifying the training process.

4.6 Visualization Analysis

In this subsection, we employ t-SNE [Van der Maaten and Hinton, 2008] for visualizing the clustering results, providing an intuitive perspective on the effectiveness of HCN in addressing the CORA and PUBMED datasets. As illustrated in Figure 6, the consensus representation derived from our approach demonstrates a significantly more compact and well-defined cluster structure. This enhanced compactness highlights HCN's superior capability to capture the intrinsic relationships and latent patterns within the data, setting it apart from the other baseline methods.

5 Conclusion

In this work, we propose a simple yet effective hypergraph clustering network, named HCN, aiming at addressing the limitations of hypergraph learning in clustering tasks. Specifically, we first employ hypergraph smoothing preprocessing to simplify the process and enhance training efficiency. Then, the preprocessed hypergraph data is passed through unshared encoders to generate different views, thereby preserving the original semantic information. Next, the self-diagonal consistency module enhances the intrinsic coherence of node representations by increasing the self-correlation of each node. Finally, to reveal the structural relationships between nodes, we adopt the structure alignment module, which employs the structural information of the hypergraph to enhance the representation learning capability. Extensive experiments on five benchmark datasets demonstrate the effectiveness of our proposed method.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant 62176203, the Fundamental Research Funds for the Central Universities (ZYTS25267, QTZX25004), and the Science and Technology Project of Xi'an (Grant 2022JH-JSYF-0009), Open Project of Anhui Provincial Key Laboratory of Multimodal Cognitive Computation, Anhui University (No. MMC202416), Selected Support Project for Scientific and Technological Activities of Returned Overseas Chinese Scholars in Shaanxi Province 2023-02, and the Xidian Innovation Fund (Project NoYJSJ25007).

References

[Antelmi *et al.*, 2023] Alessia Antelmi, Gennaro Cordasco, Mirko Polato, Vittorio Scarano, Carmine Spagnuolo, and Dingqi Yang. A survey on hypergraph representation learning. *ACM Computing Surveys*, 56(1):1–38, 2023.

[Bretto, 2013] Alain Bretto. Hypergraph theory. *An introduction. Mathematical Engineering. Cham: Springer*, 1, 2013.

[Cai *et al.*, 2022] Derun Cai, Moxian Song, Chenxi Sun, Baofeng Zhang, Shenda Hong, and Hongyan Li. Hypergraph structure learning for hypergraph neural networks. In *IJCAI*, pages 1923–1929, 2022.

[Chen *et al.*, 2024a] Man-Sheng Chen, Xi-Ran Zhu, Jia-Qi Lin, and Chang-Dong Wang. Contrastive multiview attribute graph clustering with adaptive encoders. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[Chen *et al.*, 2024b] Mulin Chen, Bocheng Wang, and Xuelong Li. Deep contrastive graph learning with clustering-oriented guidance. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 11364–11372, 2024.

[Chen *et al.*, 2025] Jianpeng Chen, Yawen Ling, Jie Xu, Yazhou Ren, Shudong Huang, Xiaorong Pu, Zhifeng Hao, Philip S Yu, and Lifang He. Variational graph generator for multiview graph clustering. *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

[Chien *et al.*, 2021] Eli Chien, Chao Pan, Jianhao Peng, and Olgica Milenkovic. You are allset: A multiset function framework for hypergraph neural networks. *arXiv preprint arXiv:2106.13264*, 2021.

[Dai and Gao, 2023] Qionghai Dai and Yue Gao. Hypergraph computation for medical and biological applications. In *Hypergraph Computation*, pages 191–221. Springer, 2023.

[Dua and Graff, 2017] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[Feng *et al.*, 2019] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3558–3565, 2019.

[Feng *et al.*, 2021] Song Feng, Emily Heath, Brett Jefferson, Cliff Joslyn, Henry Kvinge, Hugh D Mitchell, Brenda Praggastis, Amie J Eisfeld, Amy C Sims, Larissa B Thackray, et al. Hypergraph models of biological networks to identify genes critical to pathogenic viral response. *BMC bioinformatics*, 22(1):287, 2021.

[Gao *et al.*, 2022] Yue Gao, Yifan Feng, Shuyi Ji, and Rongrong Ji. Hgnn+: General hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3181–3199, 2022.

[Huang and Yang, 2021] Jing Huang and Jie Yang. Unignn: a unified framework for graph and hypergraph neural networks. *arXiv preprint arXiv:2105.00956*, 2021.

[Kim *et al.*, 2024] Sunwoo Kim, Soo Yong Lee, Yue Gao, Alessia Antelmi, Mirko Polato, and Kijung Shin. A survey on hypergraph neural networks: An in-depth and step-by-step guide. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6534–6544, 2024.

[Kingma, 2014] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Lee and Chae, 2024] Jongsoo Lee and Dong-Kyu Chae. Multi-view mixed attention for contrastive learning on hypergraphs. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2543–2547, 2024.

[Lee and Shin, 2023] Dongjin Lee and Kijung Shin. I'm me, we're us, and i'm us: Tri-directional contrastive learning on hypergraphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8456–8464, 2023.

[Li *et al.*, 2013] Dong Li, Zhiming Xu, Sheng Li, and Xin Sun. Link prediction in social networks based on hypergraph. In *Proceedings of the international conference on world wide web*, pages 41–42, 2013.

[Liu *et al.*, 2023] Yue Liu, Xihong Yang, Sihang Zhou, Xinwang Liu, Zhen Wang, Ke Liang, Wenxuan Tu, Liang Li, Jingcan Duan, and Cancan Chen. Hard sample aware network for contrastive deep graph clustering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 8914–8922, 2023.

[Liu *et al.*, 2024] Yunhui Liu, Xinyi Gao, Tieke He, Tao Zheng, Jianhua Zhao, and Hongzhi Yin. Reliable node similarity matrix guided contrastive graph clustering. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[Qian *et al.*, 2024] Yiyue Qian, Tianyi Ma, Chuxu Zhang, and Yanfang Ye. Dual-level hypergraph contrastive learning with adaptive temperature enhancement. In *Companion Proceedings of the ACM on Web Conference*, pages 859–862, 2024.

[Rossi and Ahmed, 2015] Ryan Rossi and Nesreen Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29, 2015.

[Sen *et al.*, 2008] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[Song *et al.*, 2024] Yumeng Song, Yu Gu, Tianyi Li, Jianzhong Qi, Zhenghao Liu, Christian S Jensen, and Ge Yu. Chgnn: a semi-supervised contrastive hypergraph learning network. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[Tang *et al.*, 2024] Bohan Tang, Zexi Liu, Keyue Jiang, Siheng Chen, and Xiaowen Dong. Training-free message passing for learning on hypergraphs, 2024.

[Van der Maaten and Hinton, 2008] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[Wang *et al.*, 2023] Yuxin Wang, Quan Gan, Xipeng Qiu, Xuanjing Huang, and David Wipf. From hypergraph energy functions to hypergraph neural networks. In *International Conference on Machine Learning*, pages 35605–35623. PMLR, 2023.

[Wei *et al.*, 2022] Tianxin Wei, Yuning You, Tianlong Chen, Yang Shen, Jingrui He, and Zhangyang Wang. Augmentations in hypergraph contrastive learning: Fabricated and generative. *Advances in neural information processing systems*, 35:1909–1922, 2022.

[Wu *et al.*, 2024] Hanrui Wu, Nuosi Li, Jia Zhang, Sentao Chen, Michael K Ng, and Jinyi Long. Collaborative contrastive learning for hypergraph node classification. *Pattern Recognition*, 146:109995, 2024.

[Xia *et al.*, 2021] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. Self-supervised hypergraph convolutional networks for session-based recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4503–4511, 2021.

[Xia *et al.*, 2022] Lianghao Xia, Chao Huang, Yong Xu, Jiashu Zhao, Dawei Yin, and Jimmy Huang. Hypergraph contrastive collaborative filtering. In *Proceedings of the International ACM SIGIR conference on research and development in information retrieval*, pages 70–79, 2022.

[Yang *et al.*, 2021] Yiyang Yang, Sucheng Deng, Juan Lu, Yuhong Li, Zhiguo Gong, Zhifeng Hao, et al. Graphlshc: towards large scale spectral hypergraph clustering. *Information Sciences*, 544:117–134, 2021.

[Yang *et al.*, 2023a] Xihong Yang, Yue Liu, Sihang Zhou, Siwei Wang, Wenxuan Tu, Qun Zheng, Xinwang Liu, Liming Fang, and En Zhu. Cluster-guided contrastive graph clustering network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 10834–10842, 2023.

[Yang *et al.*, 2023b] Xihong Yang, Cheng Tan, Yue Liu, Ke Liang, Siwei Wang, Sihang Zhou, Jun Xia, Stan Z Li, Xinwang Liu, and En Zhu. Convert: Contrastive graph clustering with reliable augmentation. In *Proceedings of the ACM International Conference on Multimedia*, pages 319–327, 2023.

[Yu *et al.*, 2021] Junliang Yu, Hongzhi Yin, Jundong Li, Qinyong Wang, Nguyen Quoc Viet Hung, and Xiangliang Zhang. Self-supervised multi-channel hypergraph convolutional network for social recommendation. In *Proceedings of the web conference*, pages 413–424, 2021.

[Zhou *et al.*, 2006] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. *Advances in neural information processing systems*, 19, 2006.

[Zhu *et al.*, 2018] Jianming Zhu, Junlei Zhu, Smita Ghosh, Weili Wu, and Jing Yuan. Social influence maximization in hypergraph in social networks. *IEEE Transactions on Network Science and Engineering*, 6(4):801–811, 2018.