High-order Multi-view Clustering for Generic Data
Erlin Pan, Zhao Kang

Abstract—Graph-based multi-view clustering has achieved better performance than most non-graph approaches. However, in many real-world scenarios, the graph structure of data is not given or the quality of initial graph is poor. Additionally, existing methods largely neglect the high-order neighborhood information that characterizes complex intrinsic interactions. To tackle these problems, we introduce an approach called high-order multi-view clustering (HMvC) to explore the topology structure information of generic data. Firstly, graph filtering is applied to encode structure information, which unifies the processing of attributed graph data and non-graph data in a single framework. Secondly, up to infinity-order intrinsic relationships are exploited to enrich the learned graph. Thirdly, to explore the consistent and complementary information of various views, an adaptive graph fusion mechanism is proposed to achieve a consensus graph. Comprehensive experimental results on both non-graph and attributed graph data show the superior performance of our method with respect to various state-of-the-art techniques, including some deep learning methods.

Index Terms—graph clustering, multi-view learning, high-order information, graph filtering.

I. INTRODUCTION

As one fundamental task in machine learning, clustering aims to divide a collection of unlabeled objects into multiple categories. The classical clustering methods, like $k$-means and DBSCN, are heavily dependent on the first-order relationships between samples. Specifically, they employ the pre-defined distances that measure the similarity between data points to perform the clustering. However, samples are not only similar to their neighbors but also tend to be similar to the neighbors’ neighbors. Therefore, the first-order relationships are incomplete [1] and the valuable information hidden in high-order proximity should be explored. We take two moons data as an example and show its KNN graphs ($K=5$) in different orders $n$ in Fig. 1. The bold red edge connects two neighbor nodes that belong to different classes. The number of wrong connections (red edges) and the ratio of accurate edges over whole ones are denoted by $NWE$ and $AccE$, respectively. It’s obvious that high-order graph decreases wrong relations.

To incorporate high-order interactions, numerous methods have been proposed in the past decades. They can be categorized into two classes based on the used information: topological structure-based methods and attribute-based methods. The topological structure-based methods focus on graph data and obtain high-order proximity by processing the adjacency matrix. [2] is a spectral clustering method for directed graph, demonstrating that high-order structure in the adjacency relationship is an important factor for clusters. [3] proposes a graph clustering framework that can preserve high-order structures with the help of a novel diffusion core. [4] regards high-order proximity as a random walk from one node to the other with different steps. A polynomial of adjacency matrices is applied to capture neighborhood relationships of various orders. However, the raw graph is often sparse and noisy; high-order information directly derived from the product of raw adjacency matrix accumulates errors, which could lead to inferior performance. Additionally, topological structure-based methods are not applicable to non-graph data.

Different from topological structure-based methods, which largely neglect the high-order interactions derived from the attributes of data, the attribute-based methods can handle both graph and non-graph data. One straightforward approach to obtain the high-order similarity is constructing hypergraph. [5] captures the high-order correlations among samples as well as various views via a hypergraph-induced hyper-Laplacian regularization term and a tensor factorization term. [6] obtains the second-order or high-order affinity by a score approximated by the Taylor expansion of solution at each iteration. [7] explores the high-order relationships from local and global perspectives. [8] is a robust spectral clustering method through a context-aware hypergraph similarity measure, in which the affinity information from three types of hypergraphs is combined together to explore the intrinsic interactions. Although [9, 10] show that larger hyperedges are better for hypergraph clustering, hypergraph-based methods suffer from large storage and complicated computation in the construction stage.

Recently, many practical applications involve multi-view data, which provide complementary information to help improve the task performance. Different from aforementioned methods, [11] conducts multi-view clustering through self-weighted high-order similarity fusion and utilizes a cross-view strategy to get adaptive high-order similarity. Nevertheless, the obtained similarity is not the intrinsic high-order information from data’s attributes.

To tackle above problems, we propose High-order Multi-view Clustering (HMvC), which utilizes the powers of similarity graph to capture high-order interactions among samples. The similarity graph reflects the neighbor interactions and is derived from the attribute of data. HMvC smooths attribute through graph filtering technique and learns a consensus graph from multiple views via an adaptive graph fusion mechanism. More importantly, HMvC is directly applicable to generic data with attributed features regardless of whether they contain graph or not. We summarize our contributions as follows:

This work was supported in part by the National Natural Science Foundation of China under Grant 62276053 (Corresponding author: Zhao Kang). Erlin Pan and Zhao Kang are with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China. Email: wujisixsix6@gmail.com, zkang@uestc.edu.cn
We propose a generic multi-view clustering method HMvC, which is suitable for both attributed graph data and non-graph data. Particularly, HMvC learns clustering-favorable representations through graph filtering and handles multi-view data through its adaptive graph fusion strategy.

To capture high-order information, we use the powers of similarity matrices to characterize high-order interactions. Theoretically, HMvC can obtain arbitrary-order similarity, even the infinity-order that explores all propagation paths passing similarity between samples.

The proposed method achieves promising performance on both multi-view non-graph and attributed graph datasets. Our simple method even outperforms many deep learning approaches.

II. RELATED WORK

A. Multi-view Clustering

Generally, multi-view clustering (MVC) methods focus on the global consensus and complementary information carried by multiple views to improve the performance [12, 13]. Traditional MVC methods mainly focus on non-graph data and obtain the clustering results based on attributed features. [14, 15] focus on handle incomplete multi-view clustering via learned common representations with semantic consistency from different views. And graph-based methods, like [16–18], aim to learn a consensus graph which used to obtain the clustering results from multiple attributes of data. [19] is a parameter-free auto-weighted multiple graph learning framework. Subspace-based methods find clusters via mapping the high-dimensional data points to several low-dimensional linear subspaces [20, 21] learns a joint subspace representation across all views and performs spectral clustering on the learned representation. Moreover, [22, 23] propose multi-view spectral clustering based on adaptive learning mechanism.

Attributed graph data contain both attributes and topological structures, and plentiful attributed graph MVC methods have been proposed to cluster graph nodes. [24] focuses on large-scale multi-view data and bipartite graph is imposed to approximate the similarity graph. [25] proposes a multi-view network embedding method that learns multiple relations by a unified network embedding framework. [26] is a robust multi-view spectral clustering method and uses a shared low-rank transition probability matrix derived from each single view as input to standard Markov chain method for clustering. [27] develops two methods that learn a shared graph from multiple graphs by using two weighting strategies. [28] proposes multi-view graph clustering method based on graph auto-encoder and takes into account the informative view selected by modularity. [29] is an attributed graph clustering method that exploits graph filtering and high-order neighborhood information. [30] is a GCN-based method and mainly solves graph structured data with multi-view attributes. [31] and [32] develop two methods based on contrastive learning: COMPLETER and MVGRL. COMPLETER performs data recovery and consistency learning of incomplete multi-view data simultaneously. MVGRL performs contrastive learning between two diffusion matrices transformed from the adjacency matrix to obtain clustering-favorable representation.

We can see that different methods are developed separately to tackle graph and feature data. In fact, feature and graph can complement each other. Therefore, we develop a generic framework, which can not only be applied on non-graph but also on attributed graph data through graph filtering. More importantly, our method can explore the interaction between feature and graph.

B. High-order Information Exploration

Many techniques have been developed for neighborhood information exploration ranging from the first-order to high-order relationships. For instance, [32] only preserves the first-order information while [11, 33] exploit the second-order relationships. [34] manipulates global transition matrices and combines various representations learned from multiple models, which helps to obtain a high-order information preserved graph representation. [35] proposes network embedding update method, in which the powered adjacency matrices are treated as high-order proximity matrices, and learns embeddings from the approximation of high-order proximity matrices. By applying the high-order Cheeger’s inequality, [36] develops a scalable network embedding approach, which makes the obtained embeddings capture the high-order structural information. To
take advantage of topological relationships in graph. [37] extracts arbitrary-order proximities with the help of eigen-decomposition reweighting theorem. [38] incorporates the community and high-order structures in an attempt to preserve as much structural information as possible.

However, above methods heavily depend on the adjacency matrix, which limits them to handle non-graph data. Recently, [39] proposes a novel framework to obtain the node embeddings containing abundant extrinsic and intrinsic information through optimization of a joint supervision signal designed by high-order mutual information, and it gets rid of dependence on the adjacency relationships. Differently, our method focuses on the construction of high-order similarity based on attributes, which reflects the intrinsic relationships in feature space. More importantly, our method explores the finite-order as well as the infinity-order information.

III. METHODOLOGY

A. Notations

Consider the non-graph data with \( V \) views and \( N \) data points, \( X = \{X^1, \ldots, X^V\} \) are the set of feature matrices. Each \( X^v = [x_1^v, x_2^v, \ldots, x_N^v]^{T} \in \mathbb{R}^{N \times d} \), where \( d \) is the dimension of feature. Define the attributed graph data as \( G = \{V, E_1, \ldots, E_V, X\} \), where \( V \) is the set of \( N \) nodes and \( \{E_v\}_{v=1}^{V} \) represent different types of relationships, which construct various adjacency matrices \( \{A^v\}_{v=1}^{V} \in \mathbb{R}^{N \times N} \).

B. Graph Filtering

A natural signal should be smooth on nearby nodes in term of the underlying graph. Graph filtering is employed to filter out undesirable high-frequency noise while retaining the graph geometric feature in graph signal processing [40]. To yield clean and clustering-favorable representation, we smooth features of raw data via graph filtering technique. For the \( v \)-th view of data, we first add self-loop of each node on each adjacency graph, i.e., \( \tilde{A}^v + I \). The normalized adjacency matrix \( \tilde{A}^v \) is computed by \( \tilde{A}^v = \left( \tilde{D}^v \right)^{-\frac{1}{2}} \left( \tilde{D}^v - \tilde{A}^v \right) \left( \tilde{D}^v \right)^{-\frac{1}{2}} \), where \( \tilde{D}^v \) is the corresponding degree matrix. The corresponding normalized graph laplacian \( L^v = \left( \tilde{D}^v \right)^{-\frac{1}{2}} \left( \tilde{D}^v - \tilde{A}^v \right) \left( \tilde{D}^v \right)^{-\frac{1}{2}} = I - A^v \), which can be eigen-decomposed as \( L^v = \tilde{U}^v \Lambda^v \tilde{U}^v^{T} \), where \( \Lambda^v = \text{diag} \{\lambda_1^v, \lambda_2^v, \ldots, \lambda_N^v\} \) are the eigenvalues of \( L^v \) corresponding to eigenvectors \( \tilde{U}^v = [\tilde{u}_1^v, \tilde{u}_2^v, \ldots, \tilde{u}_N^v] \). The eigenvalues are interpreted as frequencies in analogy with classical Fourier analysis. We utilize the first-order similarity introduced in section III-C as graph for non-graph data.

Then the smoothness \( \Omega(u_i^v) \) of eigenvector \( \tilde{u}_i^v \) can be measured as [41]

\[
\Omega(u_i^v) = \frac{\tilde{u}_i^v^{T} L^v \tilde{u}_i^v}{\tilde{u}_i^v^{T} \tilde{u}_i^v} = \lambda_i^v.
\]

Eq. (2) indicates that a smooth graph signal contains more low-frequency (small eigenvalues) basis signals than high-frequency ones. \( X^v \) can be decomposed into a combination of basis signals, i.e., \( X^v = \tilde{U}^v c = \sum_{i=1}^{N} \tilde{u}_i^v c_i \), where \( c_i \) is the coefficient. Moreover, the spectral response \( f(\lambda) \) of a loss-pass filter \( F \) can be defined as \( f : \lambda \mapsto f(\lambda) = 1 - \tau \lambda \). The top eigenvalue of \( L^v \) is 2 and [41] suggests that \( \tau = \frac{1}{\lambda_{\text{max}}} \) is the optimal choice. Then the filtered signal \( H^v \) is defined as

\[
H^v = \tilde{U}^v F \tilde{U}^v^{T} X^v = \tilde{U}^v F \tilde{U}^v^{T} \tilde{U}^v c
\]

\[
= \tilde{U}^v (I - \frac{1}{2} \Lambda^v) \tilde{U}^v^{T} \tilde{U}^v c
\]

\[
= (I - \frac{1}{2} L^v) X^v
\]

(2)

This can only explore the first-order neighborhood information. In practice, we use the \( k \)-th order filtering to mine high-order relation, i.e.,

\[
H^v = (I - \frac{1}{2} L^v)^k X^v.
\]

(3)

C. High-order Graph in Feature Space

Above approach is from the perspective of topological structure, here we further explore the similarity information in feature space. Similarity is a basic relationship among data points, and our model learns valuable information via making use of high-order relation. Given an attributed graph data \( G \) or generic multi-view data \( X \), we can obtain the first-order similarity of feature matrix. For the first-order similarity matrix of \( v \)-th view \( \tilde{W}_{1}^{v} \in \mathbb{R}^{N \times N} \), its element is defined as

\[
(\tilde{W}_{1}^{v})_{ij} = \frac{\cos \langle x_i^v, x_j^v \rangle}{2} + 1, \quad \text{where } \cos \langle x_i^v, x_j^v \rangle = \frac{x_i^v \cdot x_j^v}{\|x_i^v\| \|x_j^v\|}
\]

(4)

is cosine similarity between node \( i \) and node \( j \). If \( i = j \), define \( (\tilde{W}_{1}^{v})_{ij} = 0 \). Then normalized \( W_{1}^{v} \) is defined as before. Let \( \Lambda^v = \text{diag} [\lambda_1^v, \lambda_2^v, \ldots, \lambda_N^v] \) and \( U^v = [u_1^v, u_2^v, \ldots, u_N^v] \) be the eigenvalues and corresponding eigenvectors of \( W_{1}^{v} \). The high-order neighborhood relationship between data points can be obtained from high-order similarity graph. We define the \( n \)-th order graph \( W_{n}^{v} = W_{1}^{v} \otimes W_{1}^{v} \).

Lemma 1: \( W_{n}^{v}(n \geq 1) \) is a Markov matrix, i.e., \( (W_{n}^{v})_{ij} \geq 0 \) and \( \sum_{j=1}^{N} (W_{n}^{v})_{ij} = 1 \) for arbitrary \( i, j \) in the \( v \)-th view.

Proof 1: Obviously, \( W_{1}^{v} \) is a Markov matrix. \( W_{2}^{v} = W_{1}^{v} \otimes W_{1}^{v} \) and the sum of arbitrary column \( (W_{2}^{v})_{i} \) can be written as

\[
\sum_{j=1}^{N} (W_{2}^{v})_{ij} = \sum_{l=1}^{N} \left( (W_{1}^{v})_{il} (W_{1}^{v})_{lj} \right)
\]

\[
= \sum_{l=1}^{N} \left[ (W_{1}^{v})_{il} \sum_{l=1}^{N} (W_{1}^{v})_{lj} \right]
\]

\[
= \sum_{l=1}^{N} (W_{1}^{v})_{il}
\]

\[
= 1.
\]

This means that powered \( W_{1}^{v} \), like \( W_{2}^{v} \), is Markov matrix too. Because \( W_{n}^{v} \) is the product of \( n \) \( W_{1}^{v} \)'s, we obtain the same conclusion for \( n \geq 3 \).
Suppose that there are \( n \) hypothesize that the first-order and infinity-order similarity are stable relationships after moving with infinite steps. We define the mixed similarity graph as

\[
\text{the information of high-order neighborhood relationship, we require}
\]
non-zero similarity graph when computed easily:

\[
\text{and the stable matrix is represented as}
\]

\[
\sum
\]

Proof 2:

1) According to the Gershgorin circle theorem, we have the Gershgorin circle \( S_i = \{ z \in \mathbb{R} : |z - (W_n^0)_{ii}| \leq \sum_{j \neq i} |(W_n^0)_{ij}| \} \) and \( \lambda_i \in S = \bigcup_{j=1}^n S_j \) \( (i = 1, 2, \cdots, N) \). Because \( W_n^0 \) is a Markov matrix and \( W_n^1 = W_n^0 \top \), then \( S_i = \{ z \in \mathbb{R} : |z | \leq \sum_{j \neq i} |(W_n^0)_{ij}| = 1 \} \). Thus, \( |\lambda_i| \leq 1 \).

2) Define \( C^v = W_n^0 - I \), it’s clear that for all \( \forall i \in \{ 1, 2, \cdots, N \} \), \( \sum (C^v_i) = \sum_{j=1}^N c_{ij} = 0 \), where \( C^v_i \) is the \( i \)-th column of \( C^v \). For each row \( c^v_i \) of \( C^v \), we have

\[
\begin{pmatrix}
1 & 1 & \cdots & 1
\end{pmatrix}
\begin{pmatrix}
c^v_1 \\
c^v_2 \\
\vdots \\
c^v_N
\end{pmatrix} = c^v_1 + c^v_2 + \cdots + c^v_N = \sum (C^v_i), \sum (C^v_2), \cdots, \sum (C^v_N) = 0
\]

This indicates that rows of \( C^v \) are linearly dependent, so det \( (C^v) = 0 \) and \( 1 \) is one eigenvalue of \( W_n^0 \).

Lemma 3: When \( n \to \infty \), \( W_n^0 \) converges to a stable matrix and the stable matrix is represented as \( \sum_{j=1}^r u_j^v u_j^v \top \), where \( r \) is the number of eigenvalues \( \lambda^v = 1 \).

Proof 3: \( W_n^1 \) is a real symmetric matrix, then \( W_n^0 \) can be computed easily:

\[
W_n^0 = (W_n^0)^\top = \sum_{j=1}^\infty u_j^v u_j^v \top
\]

Suppose that there are \( r \) eigenvalues equal to \( 1 \), then \( W_n^0 = \sum_{j=1}^r u_j^v u_j^v \top \) when \( n \to \infty \).

Lemma 2 and 3 indicates that there exists a convergent non-zero similarity graph when \( n \to \infty \) and we obtain the infinity-order graph as \( \lim_{n \to \infty} W_n^0 = \sum_{j=1}^r u_j^v u_j^v \top \). To explore the information of high-order neighborhood relationship, we define the mixed similarity graph as

\[
f_v(W) = W_v^1 + W_v^2 + \cdots + W_v^n.
\]

Specifically, the first-order proximity is the most intuitionistic relationship among nodes and the infinity-order interactions are stable relationships after moving with infinite steps. We hypothesize that the first-order and infinity-order similarity graph contain most information, which is later verified by our ablation study. Then we define the mixed similarity graph with the least cost when \( n \to \infty \) as

\[
f_v(W) = W_v^1 + \sum_{j=1}^r u_j^v u_j^v \top.
\]

D. Graph Learning

Since the raw graph is often sparse and noisy, which is not optimal for downstream tasks, we choose to learn a refined graph from data. For multi-view data, a graph can be learned for each view based on self-expression property of data, i.e., each data point can be represented by a linear combination of other samples. The objective function on single-view data can be mathematically formulated as

\[
\min_{S^v} \| H^v \top - H^v \top S^v \|_F^2 + \alpha \| S^v - f_v(W) \|_F^2,
\]

where \( \alpha > 0 \) is a trade-off parameter and the Frobenius norm is used as a regularizer for simplicity. To incorporate high-order neighborhood relationship, we require \( S^v \) be close to the mixed similarity graph. Though some common information is shared by all views, there exists some complementary factors, such as geometry and semantics, which are distinct in different views. To obtain a consistent result, we propose a graph fusion framework, which assigns weights to various views automatically. Eventually, our proposed HMvC model can be written as

\[
\min_{S^v, \gamma} \sum_{v=1}^V \gamma^v \left( \| H^v \top - H^v \top S^v \|_F^2 + \alpha \| S^v - f_v(W) \|_F^2 \right)
\]

Suppose that there are \( r \) eigenvalues equal to \( 1 \), then \( W_n^0 = \sum_{j=1}^r u_j^v u_j^v \top \) when \( n \to \infty \).

\[
\min_{S^v, \gamma^v, U} \sum_{v=1}^V \gamma^v \left( \| H^v \top - H^v \top S^v \|_F^2 + \alpha \| S^v - f_v(W) \|_F^2 \right)
\]

where the third term is the graph fusion term that aims to obtain a unified graph \( S \) by assuming it is a linear combination of \( S^v \) from each view. \( \gamma^v \) is the weight of \( v \)-th view. If the value in the parentheses of first term is big, the corresponding \( \gamma^v \) would be small, which contributes little to final \( S \). It can be seen that there are three key components in HMvC: smoothing features via graph filtering, preserving high-order similarity information, and learning a unified graph through graph fusion mechanism. Note that graph filtering explores the high-order relation in topology structure, while high-order graph mines the high-order information in feature space. They complement each other to some extent.

\[
\min_{S^v, \gamma^v, U} \sum_{v=1}^V \gamma^v \left( \| H^v \top - H^v \top S^v \|_F^2 + \alpha \| S^v - f_v(W) \|_F^2 \right)
\]

E. Optimization Procedure

To solve problem (8), we adopt an alternating optimization strategy, in which we fix one variable at once and optimize the others alternatively.
1) Initialization of S: Firstly, we initialize S as the view average of solutions from Eq. (7).

2) Fix S and γ, update S': When S and γ are fixed, we rewrite Eq. (8) for each view as follows

$$\min_{S'} \|H^{T} - H^{T}S'\|^2_F + \alpha\|S' - f_v(W)\|^2_F +$$

$$\beta\left\|S - \sum_{i \neq v}^{V} \gamma^i S^i\right\|^2_F - \gamma^v S^v.$$

By setting its derivation w.r.t $S'$ to zero, we get

$$S' = \left(\sum_{i}^{V} \gamma^{i} S^{i}\right)^{-1} \left(\sum_{i}^{V} \gamma^{i} f_{i}(W)\right) + \beta(S - \sum_{i \neq v}^{V} \gamma^i S^i).$$

3) Fix $S'$ and $\gamma^v$, update $S$: When $S'$ and $\gamma^v$ are fixed, Eq. (8) is simplified into

$$\min_{S} \beta\left\|S - \sum_{i = 1}^{V} \gamma^i S^i\right\|^2_F + \mu\|S\|^2_F.$$ (12)

Let its derivation w.r.t S be zero, we have

$$S = \frac{\beta\sum_{i = 1}^{V} \gamma^i S^i}{\beta + \mu}.$$ (13)

4) Fix $S'$ and $S''$, update $\gamma^v$: Define $\gamma = (\gamma^1, \gamma^2, \ldots, \gamma^V)$, then our objective function is formulated as a quadratic programming (QP) problem

$$\min_{\gamma} \frac{1}{2}\gamma^{T} P \gamma + q^{T} \gamma, \quad \text{s.t.} \quad \gamma > 0, 1 \cdot \gamma = 1,$$ (14)

where $P \in \mathbb{R}^{V \times V}$ and $P_{ij} = \text{Tr}(S^i \times S^j)$, $q_v = M^v - 2 \cdot \beta \text{Tr}(S^v)$ and $S'' = \|H^{T} - H^{T}S''\|^2_F + \alpha\|S'' - f_v(W)\|^2_F$. As a standard QP problem, Eq. (14) can be solved efficiently.

**Algorithm 1 HMvC**

**Require:** parameter $\alpha$, $\beta$, $\mu$, adjacency matrix $A^1, ..., A^V$ $(W^1, ..., W^V$ for non-graph data), feature $X^1, ..., X^V$, the order of graph filtering $k$, the number of clusters $c$

**Ensure:** $c$ clusters

1. Initialize $S$ as Eq. (7) and set $\gamma^v = \frac{1}{v}$;
2. Graph filtering by Eq. (5) or Eq. (6);
3. Compute mixed high-order similarity graph $f_v(W)$ by Eq. (5) or Eq. (6);
4. while convergence condition does not meet do:
   5. Update $S''$ in Eq. (11) for each view;
   6. Update $S$ in Eq. (13);
   7. Update $\gamma^v$ via solving Eq. (14);
5. end while
6. Clustering on $S$.

The optimization procedure will monotonically decrease the objective function value in Eq. (8) in each iteration. Since the objective function has a lower bound, such as zero, the above iteration converges. The complete steps of HMvC are outlined in Algorithm 1. Afterwards, we perform classical clustering method on obtained $S$, like k-means and spectral clustering, etc., to achieve the final result.

**F. Scalable Graph Learning for Large-scale Data**

Although Eq. (8) can construct a clustering-favorable graph, it could take a lot of memory and time because the size of learned graph is $N \times N$, and Eq. (11) involves cubic computation complexity. Therefore, scalable solution is desired for large-scale data, like matrix factorization [43, 44]. One efficient approach is to select some representative samples as anchors to approximate raw data. Anchor-based strategy is mainly composed of two steps: anchor selection and the construction of anchor graph. Two popular means of anchor selection are random sampling [45] and k-means [46-48]. There are two strategies for k-means based anchor selection. [46] regards the centroid of each cluster as the landmark representation while [47, 48] choose the data points close to centroids as anchors. Differently, [49] selects anchors via a weight mechanism on the raw features of data.

However, all aforementioned anchor selection methods focus on attributed features only and neglect the topological information. Similar to [4], we choose anchors via the node importance in topological structure. Specifically, the set of anchor nodes $Y$ is chosen from the top $m$ nodes with large degree. Different from [41], we consider the mixed second-order adjacency matrix $\hat{A}^v = A^v + \tilde{A}^v$. The multi-view degree matrix is computed as $\hat{D} = \sum_{v = 1}^{V} D^v$, where $D^v$ is the mixed degree matrix of $v$-th view. Then the absolute importance of node $i$ is $g_i = \frac{d_i - \overline{d}}{\sum_{j \in V - Y} \eta g_j}$, where $\eta > 1$ is a sharpening parameter. We repeatedly select anchors via the relative importance of node

$$p_i = \frac{g_i}{\sum_{j \in V - Y} g_j}.$$ (15)

At the beginning, $Y = \emptyset$. Then we iteratively pick node $i$ with the largest $p$ in $V - Y$. After each iteration, $Y = Y \cup \{i\}$. The indexes of selected anchors are represented as $\text{Inds} = \{S_1, S_2, ..., S_{m-1}, S_m\}$, then the filtered anchor representation $\hat{H}^v$ are picked from $H^v$ by index $\text{Inds}$. The anchor-based mixed high-order graph $f_v(W)$ is constructed by Eq. (5) and Lemma 4 efficiently.

**Lemma 4**: Divide the $n$th-order similarity graph of anchors $W_n^i$ into two parts: $W_n^i = \left(\begin{array}{c}W_n^1 \vdots W_n^2 \vdots W_n^V \end{array}\right)$, $(W_n^1)^2 = \left(\begin{array}{c}(W_n^1)^1(W_n^1)^1^T \\
(W_n^1)^2(W_n^1)^2^T \end{array}\right)$ and $(W_n^2)^2$ is the remaining part, where $(W_n^1)^1 = W_n^1 | \text{Inds} |$ and $(W_n^2)^2$ is defined similarly.

**Proof 4**: The second-order similarity graph is computed by $W_2^2 = \left(\begin{array}{c}W_2^1 \vdots W_2^2 \vdots W_2^V \end{array}\right)$, and the second-order anchor-based graph is $(W_2^1)^2 = \left(\begin{array}{c}(W_2^1)^1(W_2^1)^1^T \\
(W_2^1)^2(W_2^1)^2^T \end{array}\right)$. The
rest can be defined in the same manner
\[
\begin{aligned}
(W_v^1)^{1} = \left( (W_v^2)^{1}(W_v^1)^{1\top} (W_v^2)^{1}(W_v^1)^{2\top} \right) ; \\
\vdots \\
(W_v^n)^{1} = \left( (W_v^{n-1})^1(W_v^1)^{1\top} (W_v^{n-1})^1(W_v^1)^{2\top} \right).
\end{aligned}
\]

The objective of anchor-based HMvC (AHMvC) is formulated as
\[
\min_{Z^v, \gamma^v, Z} \sum_{v=1}^{V} \gamma^v \left( \|H^v - \tilde{H}^v\|_F^2 + \alpha\|Z^v - f_v(W^v)\|_F^2 \right) + \beta \left( \|Z - \sum_{v=1}^{V} \gamma^v Z^v\|_F^2 + \mu\|Z\|_F^2 \right),
\]
\[
s.t. \sum_{v=1}^{V} \gamma^v = 1, \quad 0 \leq \gamma^v.
\]

\(Z\) or \(Z^v\) has a size of \(N \times m\), which represents the similarity between samples and anchors. The optimization strategy of AHMvC is similar to HMvC, and we show it in Algorithm 2. As a result, the computation complexity is linear to \(N\).

**Algorithm 2 AHMvC**

**Require:** parameter \(\alpha, \beta, \mu\), adjacency matrix \(\tilde{A}^1, \ldots, \tilde{A}^V\) and \(A^1, \ldots, A^V\) for non-graph data, feature \(X^1, \ldots, X^V\), the order of graph filtering \(k\), the number of clusters \(c\), anchors’ num \(m\).

**Ensure:** \(c\) clusters

1. Initialize \(S\) according to Eq. (7) and \(\gamma^v = \frac{1}{V}\);
2. Graph filtering by Eq. (8);
3. Obtain \(Inds\) of anchors by Eq. (13);
4. Select filtered anchors’ representations \(\tilde{H}^v = H^v[inds]\);
5. Construct anchor-based high-order similarity graphs via Lemma 4;
6. Continue the remaining steps of HMVC.

**TABLE I:** multi-view attributed graph datasets

| Dataset     | Nodes | Features | Graph and Edges | Clusters |
|-------------|-------|----------|-----------------|----------|
| ACM         | 3,025 | 1,830    | Co-Subject (29,281) | 3        |
|             |       |          | Co-Author (2,210,761) |          |
| DBLP        | 4,057 | 334      | Co-Paper (5,000,495) | 4        |
|             |       |          | Co-Term (6,776,335) |          |
| IMDB        | 4,780 | 1,232    | Co-Actor (98,010) | 3        |
|             |       |          | Co-Director (21,018) |          |
| Amazon photos | 7,487 | 745      | Co-Purchase(119,043) | 8        |
| Amazon computers | 13,381 | 767      | Co-Purchase(245,778) | 10       |
| AIDS        | 25,163 | 4        | relationships-1 (18,844) | 37       |
|             |       |          | relationships-2 (6,626) |          |
|             |       |          | relationships-3 (135) |          |

**TABLE II:** generic multi-view feature datasets

| View | Caltech20/Caltech7 | Citeseer |
|------|-------------------|----------|
| 1st  | Gabor(48)         | Citation Links (3,312) |
| 2nd  | Wavelet moments (40) | Words Presence (3,703) |
| 3rd  | CENTRIST (254)    |          |
| 4th  | HOG (1,984)       |          |
| 5th  | GIST (512)        |          |
| 6th  | LBP (928)         |          |

| Nodes | 2,386/1,474 | 3,312 |
| Clusters | 20/7 | 6 |

**IV. EXPERIMENT**

**A. Datasets and Metrics**

We evaluate HMvC on both multi-view attributed graph datasets and generic feature datasets. For multi-view attributed graph datasets, ACM, DBLP, IMDB [28], AIDS [50], and Amazon datasets [51] are chosen.

ACM is a paper network from ACM database. Its node features are the elements of a bag-of-words representing each paper’s keywords. Two graphs are constructed by two types of relationships: “Co-Author” means that two papers are written by the same author and “Co-Subject” suggests that they focus on the same field.

DBLP is an author network from DBLP database. Its node features are the elements of a bag-of-words representing each author’s keywords. Three graphs are derived from the relationships: "Co-Author", "Co-Conference", and "Co-Term", which indicate that two authors have worked together on the same paper, published papers at the same conference, and published papers with the same terms.

IMDB is a movie network from IMDB database. Its node features correspond to elements of a bag-of-words representing each movie. The relationships of being acted by the same actor (Co-actor) and directed by the same director (Co-director) are exploited to construct two graphs.

Amazon datasets contain Amazon photos and Amazon computers, which are segments of the Amazon co-purchase network. Their nodes represent goods and features of each good are bag-of-words of product reviews, the edge means that two goods are purchased together. To have multi-view attributes, the second feature matrix is constructed via cartesian product by following [52].

AIDS consists of graphs representing molecular compounds. Nodes are labeled with the number of corresponding chemical symbol and edges are constructed by the valence of linkage. We think the topological information is more important for AIDS than attributes because feature has only 4 dimensions.

All statistical information of datasets is shown in Table I. We adopt five popular clustering metrics, including ACCu-
racy (ACC), Normalized Mutual Information (NMI), F1 score, Adjusted Rand Index (ARI), and PURity (PUR). A higher value of these metrics indicates a better clustering performance.

B. Experimental Setup

We compare HMvC and AHMvC with several state-of-the-art multi-view graph clustering methods on graph data: graph learning-based methods, like RMSC [26], MNE [25], PwSC [27], and MVAGC [5]; GNN-based methods, like O2MA, O2MAC [28], SDCN [53], HDMI [39], DAEGA [54], CMGEC [55], and MAGCN [29]; contrastive learning-based methods, like COMPLETER [30] and MVGRL [31].

RMSC employs low-rank and sparse decomposition to achieve robust spectral clustering. MNE learns multi-view graph embedding for clustering and PwSC is a parameter-weighted multi-view graph clustering method. MVAGC is a scalable method that explores high-order interaction among topological structures. O2MA and O2MAC select the most informative view to learn representation for clustering. HDMI learns node embeddings by exploring high-order mutual information. SDCN and DAEGA are two deep clustering methods aiming to collect structural information via GCN and graph attention auto-encoder, respectively. CMGEC adds graph fusion network on multiple graph auto-encoders to obtain a consistent embedding. MAGCN applies graph auto-encoder on both attributes and topological graphs. COMPLETER and MVGRL learn a common representation shared across multiple views and different graphs via contrastive mechanism, respectively.

Furthermore, we compare HMvC with several traditional methods on generic multi-view data, including AMSL [19], MLRSSC [21], MSC_IAS [56], LMVSC [46], and SMC [57]. AMSL is a parameter-free auto-weighted multiple graph learning framework and remaining methods are subspace clustering methods.

We set $\alpha$, $\beta$, and $\mu$ in the same range $[1e^{-3}, 1, 1e2, 1e3, 1e4]$. On all datasets, we set filtering order $k = 2$. We utilize third-order similarity on graph data while second-order on non-graph data, then we tune the parameters to obtain the best results. On the medium-size datasets, we choose the number of anchors $m = [50, 100]$ for AHMvC. Almost all experiments are conducted on the same machine with Intel(R) Core(TM) i7-8700 3.20GHz CPU, two GeForce GTX 1080 Ti GPUs, and 64GB RAM. MVGRL [31] is implemented on Google Colab with Intel(R) Xeon(R) CPU 2.30GHz CPU, one Tesla K80, and 12GB RAM.

C. Experimental Results

1) Clustering task on graph data: The results on graph data are shown in Table III and Table IV. Our method achieves the best performance in most cases.

GNN can capture structural information effectively, so the GNN-based methods achieve better performance than shallow methods: MNE, PMSC, and PwMC. Among GNN-based methods, CMGEC achieves the best score in most metrics. However, HMvC improves CMGEC by almost 11%, 5%, 14% on ACC, NMI, ARI on IMDB, respectively. This is because CMGEC doesn’t explore high-order information, which will cause information loss. Compared to MVAGC, HMvC improves ACC by almost 2.1%, 0.7%, 3.2% on ACM, DBLP, IMDB, respectively. HMvC produces a little better performance than AHMvC. With respect to contrastive learning methods, our improvement is also significant. This indicates that capturing high-order relation in attributes is promising.

Our method is also efficient and consumes less memory compared to deep learning methods, which is appealing in practice. Specifically, benefiting from anchor idea, on medium-size attributed graph datasets, like Amazon and AIDS datasets, MVAGC and AHMvC are the most efficient methods. However, AHMvC improves MVAGC on ACC by more than 5% and 1% on Amazon photos, Amazon computers, respectively. On AIDS, MVAGC achieves better ACC while AHMvC is much better on all other metrics. In fact, high-order information is limited on AIDS since its feature just have 4 dimensions. From another perspective, our optimization converges fast. As observed from Fig. 2, the value of HMvC/AHMvC’s objective reaches its lower bound in less than 20 iterations.

2) Clustering task on non-graph data: All results on feature data are shown in Table V.

Although SMC is the best one among compared methods on these datasets, HMvC outperforms SMC by almost 4%, 5%, 8% on ACC on three benchmarks. The improvements of NMI and PUR are up to 8% and 5% on average. This is mainly attributed to the exploitation of high-order information. These results indicate that our approach can handle both graph and generic feature data effectively.

V. ABLATION STUDIES

A. Effect of Graph Filtering

Graph filtering is aimed to filter out undesirable high-frequency noise as well as retaining graph geometric features [4]. To show its effect in HMvC, we report the results of different $k$ on three multi-view graph datasets in Fig. 3. Red Triangles indicate the best results, while blue ones represent results without graph filtering. As $k$ increases, all results get better first but then become worse, and the best results are obtained around $k = 2$. Too large $k$ are not necessary since it will lead to over-smoothing problem, where the filtered representation becomes too smooth to distinguish.

We can see that the effects on different datasets are different. The improvement on DBLP is more than 18%, while 4% and 3% on ACM and IMDB. This is because distribution of eigenvalues varies on different datasets [58]. Specifically, there are more high-frequency components in IMDB than others, which would weaken the effectiveness of graph filter.

B. Effect of high-order neighborhood information

We set different order $n$ to explore the effect of high-order relationships. In Fig. 3, blue triangles indicate the results with the first-order graph. Caltech7 and Caltech20 achieve the best performance at $n = 2$ and $n = 5$, and ACC improvement is almost 5% with respect to $n = 1$. Therefore, the results in Tables III-V can be further improved if $n$ is tuned. On ACM, HMvC obtains better results on the third-order similarity graph,
TABLE III: The results of HMvC ($k = 2, n = 3$) and other methods. ‘OM’ means that the method raises out-of-memory problem. **Bold** numbers indicate the best score.

| Method  | ACM ACC | ACM NMI | ACM ARI | ACM F1 | DBLP ACC | DBLP NMI | DBLP ARI | DBLP F1 | IMDB ACC | IMDB NMI | IMDB ARI | IMDB F1 |
|---------|---------|---------|---------|--------|----------|----------|----------|--------|----------|----------|----------|--------|
| MNE     | 0.6370  | 0.2999  | 0.2486  | 0.6479 | OM       | OM       | OM       | OM     | 0.3958   | 0.0017   | 0.0008   | 0.3316 |
| RMSC    | 0.6315  | 0.3973  | 0.3312  | 0.5746 | 0.8994   | 0.7111   | 0.7647   | 0.8248 | 0.2702   | 0.0054   | 0.0018   | 0.3775 |
| PwMC    | 0.4162  | 0.0332  | 0.0395  | 0.3783 | 0.3253   | 0.0190   | 0.0159   | 0.2808 | 0.2453   | 0.0023   | 0.0017   | 0.3164 |
| SDCN    | 0.8631  | 0.5783  | 0.6387  | 0.8619 | 0.6497   | 0.2977   | 0.3099   | 0.6377 | 0.4047   | 0.0099   | 0.0109   | 0.3535 |
| DAEGC   | 0.8909  | 0.6430  | 0.7046  | 0.8906 | 0.8733   | 0.6742   | 0.7014   | 0.8617 | 0.3683   | 0.0055   | 0.0039   | 0.3560 |
| O2MAC   | 0.9042  | 0.6923  | 0.7394  | 0.9055 | 0.9074   | 0.7287   | 0.7780   | 0.9013 | 0.4502   | 0.0421   | 0.0564   | 0.1459 |
| HDMI    | 0.8737  | 0.6453  | 0.6736  | 0.8720 | 0.8846   | 0.6918   | 0.7530   | 0.8652 | 0.5835   | 0.1692   | 0.2033   | 0.4503 |
| CMGEC   | 0.9389  | 0.6912  | 0.7232  | 0.9072 | 0.9103   | 0.7237   | 0.7859   | 0.9042 | 0.4844   | 0.0514   | 0.0469   | 0.5101 |
| MvAGC   | 0.8975  | 0.6735  | 0.7212  | 0.8986 | 0.9277   | 0.7727   | 0.8276   | 0.9225 | 0.5633   | 0.0371   | 0.0940   | 0.3783 |
| AHMvC   | 0.9042  | 0.6923  | 0.7394  | 0.9055 | 0.9074   | 0.7287   | 0.7780   | 0.9013 | 0.4502   | 0.0421   | 0.0564   | 0.1459 |
| HDMI    | 0.8737  | 0.6453  | 0.6736  | 0.8720 | 0.8846   | 0.6918   | 0.7530   | 0.8652 | 0.5835   | 0.1692   | 0.2033   | 0.4503 |

TABLE IV: The results on medium-size datasets ($k = 2, n = 3$). Some methods are removed due to OM.

| Datasets       | Method       | ACC | NMI | ARI | F1 | Time(s) |
|----------------|--------------|-----|-----|-----|----|---------|
| Amazon Photos  | COMPLETER    | 0.3678 | 0.2606 | 0.0759 | 0.3067 | 421.3 |
|                | MVGRL        | 0.5054 | 0.4331 | 0.2379 | 0.4599 | 994.2 |
|                | MAGCN        | 0.5167 | 0.3897 | 0.2401 | 0.4736 | 3783.6 |
|                | MvAGC        | 0.6775 | 0.5237 | 0.3968 | 0.6397 | 37.8  |
|                | HMvC         | 0.7332 | 0.6504 | 0.5788 | 0.7188 | 60.5  |
| Amazon Computers | COMPLETER   | 0.2417 | 0.1562 | 0.0536 | 0.1601 | 844   |
|                | MVGRL        | 0.2450 | 0.1012 | 0.0553 | 0.1706 | 1520  |
|                | MvAGC        | 0.5796 | 0.3957 | 0.3224 | 0.4117 | 164.8 |
|                | AHMvC        | 0.5904 | 0.5298 | 0.3729 | 0.4892 | 199.7 |
| AIDS           | MvAGC        | 0.5118 | 0.0700 | 0.0182 | 0.0364 | 847.9 |
|                | AHMvC        | 0.2629 | 0.2799 | 0.0661 | 0.1714 | 752.5 |

Fig. 2: The objective variation on ACM, Amazon photos and Amazon computers.

TABLE V: The results on feature data ($k = 2, n = 2$).

| Datasets       | Metrics   | AMSL ACC | AMSL NMI | AMSL ARI | AMSL F1 | LMVSC ACC | LMVSC NMI | LMVSC ARI | LMVSC F1 | MLRSSC ACC | MLRSSC NMI | MLRSSC ARI | MLRSSC F1 | MSC IAS ACC | MSC IAS NMI | MSC IAS ARI | MSC IAS F1 | SMC ACC | SMC NMI | SMC ARI | SMC F1 | HMvC ACC | HMvC NMI | HMvC ARI | HMvC F1 |
|----------------|-----------|----------|----------|----------|--------|----------|----------|----------|--------|----------|----------|----------|----------|--------|-------------|-------------|-------------|-------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Caltech7       | ACC       | 0.4518   | 0.7266   | 0.3731   | 0.3976 | 0.7869   | 0.8215   | 0.8398   | 0.5969 |
|                | NMI       | 0.4243   | 0.5193   | 0.2111   | 0.2455 | 0.4829   | 0.5696   | 0.5600   | 0.6419 |
|                | PUR       | 0.4674   | 0.7517   | 0.4145   | 0.4444 | 0.8860   | 0.8398   | 0.7225   | 0.3707 |
| Caltech20      | ACC       | 0.3013   | 0.5306   | 0.3731   | 0.3127 | 0.5716   | 0.6249   | 0.5458   | 0.6327 |
|                | NMI       | 0.4054   | 0.2571   | 0.2111   | 0.2455 | 0.4829   | 0.5696   | 0.5600   | 0.6419 |
|                | PUR       | 0.3164   | 0.5847   | 0.4145   | 0.3374 | 0.6266   | 0.7225   | 0.5600   | 0.6542 |
| Citeseer       | ACC       | 0.1687   | 0.5226   | 0.2509   | 0.3411 | 0.5600   | 0.6419   | 0.5600   | 0.6542 |
|                | NMI       | 0.0023   | 0.2571   | 0.0267   | 0.1153 | 0.2985   | 0.3707   | 0.2985   | 0.3707 |
|                | PUR       | 0.1687   | 0.5446   | 0.6370   | 0.8076 | 0.5600   | 0.6542   | 0.5600   | 0.6542 |
and the improvement of ACC is up to 2.1% compared with the first-order. It also benefit from the infinity-order graph. Although exploring high-order neighborhood information with a larger $n$ may not achieve better performance because there could be too much extraneous noisy information, the result of high-order similarity is often better than that of the first-order one. The suitable value of $n$ is chosen less than 5 or the infinite value empirically.

In Fig. 5, we demonstrate the average of change rate, i.e., $\left| (W^n_{ij} - W^{n-1}_{ij}) / (W^{n-1}_{ij}) \right|$, in similarity matrices between different orders on Citeseer, DBLP, Caltech7, and Caltech20. Clearly, there exists u-shape and two big changes on all results. The first one occurs between first-order and second-order, which means the information of these two matrices are much different. The big change between the finite-order and infinity-order matrix suggests that the infinity-order similarity contain some unique information. There is little change between other orders. This analysis validates that the second-order and infinity-order similarity matrices contain much valuable information.

C. Parameter Analysis

There are three trade-off parameters in our method, $\alpha$, $\beta$, and $\mu$. Fig. 6 shows sensitivity analysis of three parameters on Caltech7. We fix one parameter and vary others in each figure. We find too large or too small $\alpha$ results in sub-optimal performance. When fix $\alpha$ at a suitable value, we find results change little if $\beta$ or $\mu$ varies. However, too large $\beta$ or $\mu$ weakens high-order graph part and results in inferior performance.

VI. Conclusion

In this paper, we develop a generic multi-view clustering method, which effectively explores the high-order information in both topology and feature space. To obtain a smooth representation and unify the process of graph and non-graph data, we apply graph filtering technique to handle the raw features. To capture high-order feature information, we define the high-order and infinity-order graphs by leveraging the powers of similarity matrix and convergent matrix. To solve the scalability issue on medium-size or large-scale dataset, we use the anchor idea to construct scalable high-order graphs. Our approach achieves promising performance on generic data. Therefore, it is beneficial to treat non-graph data from the perspective of graph. In addition, the proposed infinity graph is general to be broadly applied in many other machine learning tasks. However, one potential limitation of our work is that the graph construction consumes time and space. In future work,
we plan to improve the graph filter and construction strategy of one-time similarity graph.

REFERENCES

[1] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, “Line: Large-scale information network embedding,” in Proceedings of the 24th international conference on world wide web, 2015, pp. 1067–1077.

[2] S. Laenen and H. Sun, “Higher-order spectral clustering of directed graphs,” in Advances in Neural Information Processing Systems, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 941–951.

[3] D. Zhou, S. Zhang, M. Y. Yildirim, S. Alcorn, H. Tong, H. Davulcu, and J. He, “High-order structure exploration on massive graphs: A local graph clustering perspective,” ACM Transactions on Knowledge Discovery from Data (TKDD), vol. 15, no. 2, pp. 1–26, 2021.

[4] Z. Lin and Z. Kang, “Graph filter-based multi-view attributed graph clustering,” in Proceedings of the 30th International Joint Conference on Artificial Intelligence, IJCAI, 2021, pp. 19–26.

[5] Z. Li, C. Tang, X. Zheng, X. Liu, W. Zhang, and E. Zhu, “High-order correlation preserved incomplete multi-view subspace clustering,” IEEE Transactions on Image Processing, vol. 31, pp. 2067–2080, 2022.

[6] M. Leordeanu and C. Sminchisescu, “Efficient hypergraph clustering,” in Artificial Intelligence and Statistics. PMLR, 2012, pp. 676–684.

[7] Y. Takai, A. Miyauchi, M. Ikeda, and Y. Yoshida, “Hypergraph clustering based on pagerank,” in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1970–1978.

[8] X. Li, W. Hu, C. Shen, A. Dick, and Z. Zhang, “Context-aware hypergraph construction for robust spectral clustering,” IEEE Transactions on Knowledge and Data Engineering, vol. 26, no. 10, pp. 2588–2597, 2013.

[9] G. Zhong and C.-M. Pun, “Revisiting nyström extension for hypergraph clustering,” Neurocomputing, vol. 403, pp. 247–256, 2020.

[10] P. Purkait, T.-J. Chin, A. Sadri, and D. Suter, “Clustering with hypergraphs: the case for large hyperedges,” IEEE transactions on pattern analysis and machine intelligence, vol. 39, no. 9, pp. 1697–1711, 2016.

[11] H. Peng, H. Wang, Y. Hu, W. Zhou, and H. Cai, “Multi-dimensional clustering through fusion of high-order similarities,” Pattern Recognition, vol. 121, p. 108108, 2022.

[12] Y. Liang, D. Huang, C.-D. Wang, and S. Y. Philip, “Multi-view graph learning by joint modeling of consistency and inconsistency,” IEEE Transactions on Neural Networks
and Learning Systems, 2022.

[13] F. Sun, X. Xie, J. Qian, Y. Xin, Y. Li, C. Wang, and G. Chao, “Multi-view k-proximal plane clustering,” Applied Intelligence, pp. 1–15, 2022.

[14] J. Wen, Z. Zhang, Z. Zhang, L. Fei, and M. Wang, “Generalized incomplete multiview clustering with flexible locality structure diffusion,” IEEE transactions on cybernetics, vol. 51, no. 1, pp. 101–114, 2020.

[15] H. Tang and Y. Liu, “Deep safe incomplete multi-view clustering: Theorem and algorithm,” in International Conference on Machine Learning. PMLR, 2022, pp. 21090–21110.

[16] K. Zhan, F. Nie, J. Wang, and Y. Yang, “Multiview consensus graph clustering,” IEEE Transactions on Image Processing, vol. 28, no. 3, pp. 1261–1270, 2018.

[17] R. Wang, F. Nie, Z. Wang, H. Hu, and X. Li, “Parameter-free weighted multi-view projected clustering with structured graph learning,” IEEE Transactions on Knowledge and Data Engineering, vol. 32, no. 10, pp. 2014–2025, 2019.

[18] H. Wang, Y. Yang, and B. Liu, “Gmc: Graph-based multi-view clustering,” IEEE Transactions on Knowledge and Data Engineering, vol. 32, no. 6, pp. 1116–1129, 2019.

[19] F. Nie, J. Li, and X. Li, “Parameter-free auto-weighted multiple graph learning: a framework for multiview clustering and semi-supervised classification,” in Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, 2016, pp. 1881–1887.

[20] M.-S. Chen, L. Huang, C.-D. Wang, D. Huang, and S. Y. Philip, “Multiview subspace clustering with grouping effect,” IEEE Transactions on Cybernetics, 2020.

[21] M. Brbić and I. Kopriva, “Multi-view low-rank sparse subspace clustering,” Pattern Recognition, vol. 73, pp. 247–258, 2018.

[22] D. Shi, L. Zhu, J. Li, Z. Cheng, and Z. Zhang, “Flexible multiview spectral clustering with self-adaptation,” IEEE Transactions on Cybernetics, 2021.

[23] Y. Chen, X. Xiao, Z. Hua, and Y. Zhou, “Adaptive transition probability matrix learning for multiview spectral clustering,” IEEE Transactions on Neural Networks and Learning Systems, 2021.

[24] Y. Li, F. Nie, H. Huang, and J. Huang, “Large-scale multi-view spectral clustering via bipartite graph,” in Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015.

[25] H. Zhang, L. Qiu, L. Yi, and Y. Song, “Scalable multiplex network embedding,” in IJCAI, vol. 18, 2018, pp. 3082–3088.

[26] R. Xia, Y. Pan, L. Du, and J. Yin, “Robust multi-view spectral clustering via low-rank and sparse decomposition,” in Proceedings of the AAAI conference on artificial intelligence, vol. 28, no. 1, 2014.

[27] F. Nie, J. Li, X. Li et al., “Self-weighted multiview clustering with multiple graphs.” in IJCAI, 2017, pp. 2564–2570.

[28] S. Fan, X. Wang, C. Shi, E. Lu, K. Lin, and B. Wang, “One2multi graph autoencoder for multi-view graph clustering,” in Proceedings of The Web Conference 2020, 2020, pp. 3070–3076.

[29] Y. Chen, S. Wang, F. Zheng, and Y. Cen, “Graph-regularized least squares regression for multi-view subspace clustering,” Knowledge-Based Systems, vol. 194, p. 105482, 2020.

[30] Y. Lin, Y. Gou, Z. Liu, B. Li, J. Lv, and X. Peng, “Completer: Incomplete multi-view clustering via contrastive prediction,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2021.

[31] K. Hassani and A. H. Khasahmadi, “Contrastive multi-view representation learning on graphs,” in International Conference on Machine Learning. PMLR, 2020, pp. 4116–4126.

[32] M. Qu, J. Tang, J. Shang, X. Ren, M. Zhang, and J. Han, “An attention-based collaboration framework for multi-view network representation learning,” in Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 1767–1776.

[33] D. Wang, P. Cui, and W. Zhu, “Structural deep network embedding,” in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 1225–1234.

[34] S. Cao, W. Lu, and Q. Xu, “Grarep: Learning graph representations with global structural information,” in Proceedings of the 24th ACM international on conference on information and knowledge management, 2015, pp. 891–900.

[35] C. Yang, M. Sun, Z. Liu, and C. Tu, “Fast network embedding enhancement via high order proximity approximation,” in Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 3894–3900.

[36] J. Zhang, Y. Dong, Y. Wang, J. Tang, and M. Ding, “Prone: Fast and scalable network representation learning,” in Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019. IJCAI, 2019, pp. 4278–4284.

[37] Z. Zhang, P. Cui, X. Wang, J. Pei, X. Yao, and W. Zhu, “Arbitrary-order proximity preserved network embedding,” in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 2778–2786.

[38] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, “Community preserving network embedding,” in Thirty-first AAAI conference on artificial intelligence, 2017.

[39] B. Jing, C. Park, and H. Tong, “Hdmi: High-order deep multiplex infomax,” in Proceedings of the Web Conference 2021, 2021, pp. 2441–2424.

[40] H. E. Egilmez, E. Pavez, and A. Ortega, “Graph learning from filtered signals: Graph system and diffusion kernel identification,” IEEE Transactions on Signal and Information Processing over Networks, vol. 5, no. 2, pp. 360–374, 2018.

[41] G. Cui, J. Zhou, C. Yang, and Z. Liu, “Adaptive graph encoder for attributed graph embedding,” in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 976–985.
[42] J. C. Bezdek and R. J. Hathaway, “Convergence of alternating optimization,” *Neural, Parallel & Scientific Computations*, vol. 11, no. 4, pp. 351–368, 2003.

[43] J. Fan, “Large-scale subspace clustering via k-factorization,” in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 342–352.

[44] F. N. De Wang and H. Huang, “Fast robust non-negative matrix factorization for large-scale human action data clustering,” 2016.

[45] Q. Wang, Y. Miao, M. Chen, and X. Li, “Spatial-spectral clustering with anchor graph for hyperspectral image,” *arXiv preprint arXiv:2104.11904*, 2021.

[46] Z. Kang, W. Zhou, Z. Zhao, J. Shao, M. Han, and Z. Xu, “Large-scale multi-view subspace clustering in linear time,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 4412–4419.

[47] M. Wang, W. Fu, S. Hao, D. Tao, and X. Wu, “Scalable semi-supervised learning by efficient anchor graph regularization,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 7, pp. 1864–1877, 2016.

[48] E. Pan received the B.Sc. degree in computer science and technology in 2021 and is currently studying for a M.S degree with School of Computer Science and Engineering, the University of Electronic Science and Technology of China, Chengdu, China. His main research interests include graph learning, multi-view learning, and clustering.

Zhao Kang received the Ph.D. degree in computer science from Southern Illinois University Carbondale, Carbondale, IL, USA, in 2017. He is currently an Associate Professor with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China. He has published over 80 research papers in top-tier conferences and journals, including NeurIPS, AAAI, IJCAI, ICDE, CVPR, SIGKDD, ECCV, ICDM, CIKM, SDM, IEEE TRANSACTIONS ON CYBERNETICS, IEEE Transactions on Image Processing, IEEE Transactions on Knowledge and Data Engineering, and IEEE Transactions on Neural Networks and Learning Systems. His research interests are machine learning, pattern recognition, and data mining. Dr. Kang has been an AC/SPC/PC Member or a Reviewer for a number of top conferences, such as NeurIPS, ICLR, AAAI, IJCAI, CVPR, ICCV, MM, and ECCV. He regularly serves as a Reviewer for the Journal of Machine Learning Research, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and IEEE TRANSACTIONS ON MULTIMEDIA.