Consistent Multiple Graph Embedding for Multi-View Clustering

Yiming Wang, Dongxia Chang, Zhiqiang Fu, and Yao Zhao, Senior Member, IEEE

Abstract—Graph-based multi-view clustering aiming to obtain a partition of data across multiple views, has received considerable attention in recent years. Although great efforts have been made for graph-based multi-view clustering, it is still challenging to fuse characteristics from various views to learn a common representation for clustering. In this paper, we propose a novel Consistent Multiple Graph Embedding Clustering framework (CMGEC). Specifically, a multiple graph auto-encoder (M-GAE) is designed to flexibly encode the complementary information of multi-view data using a multi-graph attention fusion encoder. To guide the learned common representation maintaining the similarity of the neighboring characteristics in each view, a Multi-view Mutual Information Maximization module (MMIM) is introduced. Furthermore, a graph fusion network (GFN) is devised to explore the relationship among graphs from different views and provide a common consensus graph needed in M-GAE. By jointly training these models, the common representation can be obtained, which encodes more complementary information from multiple views and depicts data more comprehensively. Experiments on three types of multi-view datasets demonstrate CMGEC outperforms the state-of-the-art clustering methods.

Index Terms—Graph neural networks, multi-view clustering, mutual information, representation learning.

I. INTRODUCTION

With the advance of information technology, multiple views of objects can be readily acquired in many domains. For instance, a piece of news can be reported by multiple news organizations, and an image can be described in different features: GIST, SIFT, and HOG, etc. Multi-view data can provide more comprehensive characteristics and helpful information than single-view [1], [2]. With the advent of multi-view data, many multi-view clustering methods [3] have emerged and are widely applied in medicine [4], and computer vision [5], etc. For example, Chao et al. [4] propose a multi-view co-clustering algorithm and apply the algorithm to an opioid dependence treatment study. However, there are still some challenges to multi-view clustering. Large differences between data from different views may produce view disagreement, which can distort a similarity matrix used to depict samples within the same class [6], [7]. Additionally, the dimension difference of different features can lead to difficulties in feature fusion [8].

Various multi-view clustering methods have been proposed to solve the above problems, including graph-based methods [9], [10], subspace-based methods [11], kernel-based methods [12], [13], etc. The graph-based multi-view clustering methods [14], [15] seek to find a fusion graph across all views and use graph-cut algorithms or other technologies to produce the clustering assignments. This kind of method can solve the problem of dimensional differences of different views. However, graph-based multi-view clustering methods are generally shallow models with limited capacity to reveal the relations in complex multi-view data. Moreover, these models can hardly combine graph structural information with data intrinsic characteristics, which are equally important for clustering tasks.

Graph Convolutional Networks (GCN) [16], [17] recently emerged can encode both the graph structure and node characteristics for latent node representation. The GCN follows a message-passing manner that aggregates a node feature information from its topological neighbors in each convolutional layer. It is consistent with the clustering task of aggregating similar samples into clusters. Thus, many GCN-based clustering methods [18], [19] have been proposed. For instance, to jointly integrate the information from node content and consensus graph, [20] employs graph representation learning techniques to ensemble clustering. To achieve mutual benefit for both learned embedding and graph clustering, Wang et al. [18] propose a goal-directed graph attention autoencoder for attributed graph clustering. These GCN-based methods greatly improve the performance of graph-based clustering. However, all these methods can handle single view data, and there are few GCN-based multi-view clustering algorithms. To model multi-view graph information, Fan et al. [21] employ one informative graph view to reconstruct multiple graph views to capture the shared representation of multiple graphs. Unlike the general multi-view clustering method applied to multi-view data and graphs, it is employed for single-view data with multi-view graphs. The above methods apply GCN to exploit both graph structure and node content to learn a latent representation. However, the graph in most GCN-based methods is fixed, making the clustering performance heavily dependent on the predefined graph. And a noisy graph with unreliable connections can
result in ineffective convolution with wrong neighbors on the graph [22], which may worsen graph clustering performance.

In order to solve the above challenges in multi-view clustering, we propose a Consistent Multiple Graph Embedding Clustering framework (CMGE), which is mainly composed of Multiple Graph Auto-Encoder (M-GAE), Multi-view Mutual Information Maximization module (MMIM) and Graph Fusion Network (GFN). Our major contributions can be summarized as follows:

- To capture the complementary information and internal relations of each view well, we propose a multi-graph attention fusion encoder to adaptively learn a common representation from multiple views.
- To maintain consistency within views, multi-view mutual information maximization is devised to make similar instances still similar to each other in the common space.
- To explore the relationships among different view graphs, a graph fusion network is devised to fuse graphs from multiple views to get a consensus graph needed in the multiple graph auto-encoder. And to improve the separability of the consensus graph, the rank constraint on its Laplacian matrix is utilized to train the GFN.
- We have conducted experiments on three types of multi-view data, and experiments show that our CMGE outperforms state-of-the-art clustering methods.

II. RELATED WORKS

Before introducing the proposed CMGE, multi-view clustering and mutual information maximization are briefly introduced in this section.

1) Multi-View Clustering: Recently, multi-view learning has attracted lots of attention, and numerous multi-view clustering methods have been proposed. We roughly divide them into three categories: graph-based methods [23], kernel-based methods [13], [24], and subspace-based methods [11]. Subspace-based multi-view clustering uses subspace learning to obtain a latent subspace shared by different views, which solves the difficulty of handling high-dimensional data to some extent. For example, to alleviate the problem that most subspace-based methods are heavily influenced by the original features, [11] explores underlying complementary information from multiple views and simultaneously seeks the underlying latent representation. Introducing neural networks, Li et al. [25] construct subspace representations linked with a latent representation to identify the underlying cluster structure of high-dimensional data. Kernel-based methods typically combine a set of base kernels constructed in different views to obtain the clustering results. To alleviate the problem of the high computational complexity of kernel-based clustering, Wang et al. [12] first maximize alignment between consensus clustering matrix and weighted base partitions. To further reduce storage and computational complexity, [13] jointly learns a consensus clustering matrix, imputes each incomplete base matrix, and optimizes the corresponding permutation matrices. These multi-view methods show satisfactory performance but suffer from two main disadvantages: (a) They can typically employ one of graph structure or data characteristics; (b) Subspace-based multi-view methods are typically sensitive to initialization; (c) Kernel-based multi-view methods suffer from intensive computational complexities.

For the graph-based multi-view clustering methods, the view-specific graphs are constructed based on the $k$-NN graph and used to find a fusion graph across all views. Most graph-based multi-view clustering methods are based on spectral clustering, a classic data clustering algorithm aiming to build a normalized affinity matrix and compute the eigenvectors of this normalized affinity matrix. Combined with graph fusion, it can be extended to multi-view clustering. Based on spectral partitioning and local refinement, Chikhi [26] presents a parameter-free multi-view spectral clustering algorithm. To address the issue that dependencies among views often delude correct predictions, Son et al. [27] propose a spectral clustering method to deal with multi-view data and dependencies among views based on the brainstorming process. Nie et al. [9] propose a Laplacian rank constrained graph, which can be approximate as the centroid of the built graph for each view with different confidences. In order to solve the problem that graph-based clustering highly depends on the quality of a predefined graph, [28] learns a global graph, which has an exact number of the connected components that reflects cluster indicators. To sufficiently consider weights of different views, Wang et al. [23] propose a graph-based multi-view clustering (GMC) method coupling the learning of the similarity-induced graphs, the unified graph, and the clustering task into a joint clustering framework. However, one major drawback of these shallow models is that they have limited capacity to reveal the deep relations in complex graph data.

2) Mutual Information Maximization: To maintain the consistency of similar samples in each view, we employ mutual information maximization in our model. Mutual information is a Shannon entropy-based fundamental quantity for measuring the relationship between random variables [29]. Following regularized information maximization (RIM) [30], maximizing the mutual information between input samples and latent cluster assignments can be used in discriminative clustering. Some deep clustering methods further study this concept [29], [31], which learn discriminative neural network classifiers that maximize the mutual information. Furthermore, mutual information is also widely used in multi-view learning. In particular, [32] proposes a self-supervised representation learning based on maximizing mutual information between features extracted from multiple views of a shared context. Mao et al. [33] propose deep mutual information maximum (DMIM) for cross-modal clustering, which preserves the shared multi-view information while eliminating the superfluous information of individual modalities in an end-to-end manner. Generally, mutual information maximization corresponds to maximizing the following objective:

$$I(X, K) = H(K) - H(K|X)$$ (1)

where $H(\cdot)$ and $H(\cdot|\cdot)$ are the entropy and conditional entropy, respectively. $K \in \{1, \ldots, K\}$ and $X \in X$ denote random variables for cluster assignments and data samples, respectively. And
the mutual information between sample $X$ and latent representation $Z$ can be understood as:

$$I(X, Z) = \int \int p(z|x)p(x) \log \frac{p(z|x)}{p(z)} dx dz = KL(p(z|x)p(x)||p(z)p(x))$$

where $p(x)$ is the distribution of the input samples and $p(z|x)$ is the distribution of the latent representations. The distribution of the latent space $p(z)$ can be calculated by $p(z) = \int p(z|x)p(x)dx$. And adversarial learning can be used to constrain the latent representations to have desired statistical characteristics specific to the input samples.

III. THE PROPOSED MODEL

As aforementioned, current graph-based multi-view clustering methods have the following shortcomings: (a) The shallow model can hardly combine the graph structural information with the node intrinsic characteristics; (b) The GCN-based methods generally use a fixed graph structure, and its performance depends heavily on the predefined graph. To handle these two challenges, CMGEC is proposed, and the overall framework is shown in Fig. 1. Our CMGEC mainly contains three parts: M-GAE, MMIM, and GFN. Firstly, the predefined graph of each view is input into the GFN to obtain the consensus graph. In order to make the consensus graph more sparse and cluster-friendly, the rank constraint on its Laplacian matrix is used to train the GFN. Then the raw features, the graph of each view, and the consensus graph are fed into the M-GAE to learn a common latent representation. To flexibly incorporate information from all views, a multi-graph attention fusion encoder is introduced into the M-GAE. Moreover, MMIM is devised to make the learned common representation maintain the similarity of the neighboring characteristics. In the following, we will describe our proposed model in detail.

A. Multiple Graph Auto-Encoder

In order to represent both multi-view graph structure and node feature comprehensively in a unified framework, we develop an M-GAE in which a multi-graph attention fusion encoder learns common latent representation. Moreover, the view-specific decoders are designed to reconstruct multi-view graph data from the learned representation.

1) Multi-Graph Attention Fusion Encoder: To learn a common representation that can fully integrate information from multiple views, a multi-graph attention fusion layer, which can fuse multi-view data and graphs adaptively, is devised based on the GCN [16]. After getting the common representation, the final representation $Z$ can be obtained by the GCN using the common representation and graph. Here, the common graph is obtained by GFN, and we will introduce it in detail in the next section.

The GCN extends the operation of convolution to graph data in the spectral domain. Here, $Z(l)$ is the representation learned by the $l$-th layer of GCN, and it can be obtained by the following

Formally, given a multi-view dataset $X = \{X^{(v)}\}_{v=1}^{V}$, consisting of $N$ samples from $V$ views, $X^{(v)} \in \mathbb{R}^{N \times d_{v}}$ denotes the feature matrix of the $v$-th view, $d_{v}$ is the dimension of the feature of the $v$-th view. $A = \{A^{(v)}\}_{v=1}^{V} \in \mathbb{R}^{N \times N}$ represents the graph of each view, $A^{(v)}$ denotes the common graph, where $A^{(v)} = f(A; \theta_{g}) \in \mathbb{R}^{N \times N}$ is learned by the GFN. The parameters of the GFN are defined by $\theta_{g}$, $Z$ denotes the common latent representation, where $Z = f(X, A, A^{(1)}; \theta_{g}) \in \mathbb{R}^{N \times m}$ is learned by the multi-graph fusion encoder $E$. The parameters of the encoder are defined by $\theta_{e}$, and $m$ is the dimension of the learned common representation. $\{\tilde{A}^{(v)}\}_{v=1}^{V} = f(Z; \theta_{d}) \in \mathbb{R}^{m \times m}$ represents the reconstructed graph relation, which is the output of the view-specific decoders $D$, and the parameters of the decoders are denoted by $\theta_{d}$.
In order to flexibly integrate the view-specific representation, a multi-graph attention fusion layer is devised using the attention strategy. Unlike graph attention networks that learn hidden representations of each node by weighting the representations for each nearest neighbor in the same view, our approach focuses on the weighting of different views. To adaptively fuse the representation of a sample in different views, attention coefficient matrix $W_a$ is introduced to learn the importance of the edge in different views. Hence, the common representation learned by the multi-graph attention fusion layer can be obtained by the following operation:

$$Z(v) = \phi \left( (\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}) Z_{(l-1)} W(l) \right)$$

where $\tilde{A} = A + I$ and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. $I$ is the identity diagonal matrix, $W(l)$ denotes the learned parameter matrix, and $\phi(\cdot)$ is an activation function.

### View-Specific Graph Decoders

In order to flexibly integrate the view-specific representation, a multi-graph attention fusion layer is devised using the attention strategy. Unlike graph attention networks that learn hidden representations of each node by weighting the representations for each nearest neighbor in the same view, our approach focuses on the weighting of different views. To adaptively fuse the representation of a sample in different views, attention coefficient matrix $W_a$ is introduced to learn the importance of the edge in different views. Hence, the common representation learned by the multi-graph attention fusion layer can be obtained by the following operation:

$$Z(v) = \phi \left( (\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}) Z(v) W(v) \right)$$

where $\tilde{A} = A + I$ and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. $I$ is the identity diagonal matrix, $W(v)$ denotes the learned parameter matrix in the $v$-th view-specific decoder.

### Reconstruction Loss

To train the M-GAE, we minimize the sum of reconstruction error of each view by measuring the difference between $A(v)$ and $\hat{A}(v)$.

$$L_{rec} = \sum_{v=1}^{V} L_{rec}^{(v)} = \sum_{v=1}^{V} \sum_{v=1}^{V} \text{loss}(A(v), \hat{A}(v))$$

where $L_{rec}^{(v)}$ is the reconstruction loss for the $v$-th view and $L_{rec}$ is the reconstruction loss for all views.

### Graph Fusion Network

Multi-view data and graphs provide multiple independent and complementary information from multiple feature spaces, and their analysis can often result in more integrated and accurate results than single view [34]. However, each graph contains different adjacency relations in different views and cannot be used directly in the common space. Therefore, to explore relationships among different views and provide global node relationships, a graph fusion network is devised to produce the consensus graph $A^*$.

In our model, a fully connected network is employed to learn the consensus graph. Specifically, the consensus graph learned by the $l$-th layer in the graph fusion network can be described as:

$$G(l) = \phi(W_{g(l)} G(l-1) + b_{g(l)})$$

where $\phi$ is the activation function of the fully connected layers, $W_{g(l)}$ and $b_{g(l)}$ are the weight matrix and bias of the $l$-th layer in the graph fusion network, respectively. To adaptively fuse each graph, a multi-graph fusion layer is placed on the first layer of GFN, and it is defined as:

$$G(1) = \phi \left( \sum_{v=1}^{V} W_f(W^{(v)} g^{(v)} A^v + b^{(v)} g^{(v)}) \right)$$

where $W_f$ is the attention coefficient matrix that indicates the importance of the edge in different views.

In order to combine the features of each graph and make the consensus graph $A^*$ more suitable for clustering, the loss of the graph fusion network is defined as

$$L_G = L_{gre} + \lambda_1 L_{gtr}$$

$$= \sum_{v=1}^{V} \text{loss}(A(v), A^*) + \lambda_1 tr(Q^T L_{A*,}\text{tr})$$

s.t. $Q^T Q = I$

where $L_{gre}$ and $L_{gtr}$ are the graph reconstruction loss for all views and the Ratio Cut used spectral clustering, respectively. $tr(\cdot)$ is the trace operator. $L_{A*}$ is the Laplacian matrix of $A^*$ and $Q$ is the relaxed indicator which can be computed by the eigenvalue decomposition of $L_{A*}$. $\lambda_1$ is a hyperparameter that balances these two losses.

Obviously, the consensus graph $A^*$ can be segmented directly to obtain the clustering results. Since the GFN uses only the graph structure and ignores the intrinsic characteristics of nodes, the clustering result obtained using $A^*$ is worse than that of the learned common representation.

### Multi-View Mutual Information Maximization

In the fields of subspace learning, it has been well recognized that if two samples are close to each other, their corresponding low dimensional representations should also be close in the latent space [35]. Therefore, the inter-sample information can be used to guide the autoencoder to learn more cluster-friendly representation.
is a discriminator, and the representation of sample and $p$ are used to replace $x$ and $T(x)$. 

$$x^2 = (14) = -\rho q$$

In our loss function, and $x^2$ and $z$ are defined as 

$$\rho = 100Leaves, and Cub; IMDB, N are adjacent. + \lambda (b) (18)$$

For convenience, these datasets are summarized in Table I, II and III.

$$\text{TABLE I}$$

| Datasets          | Classes | Nodes | Feature1 | Feature2 | Feature3 | Feature4 |
|-------------------|---------|-------|----------|----------|----------|----------|
| 3source           | 6       | 169   | 3560     | 3631     | 3068     | -        |
| BBC               | 5       | 685   | 4659     | 4633     | 4665     | 4684     |
| 100LEAVES         | 100     | 1600  | 64       | 64       | 64       | -        |
| Cub               | 10      | 600   | 1024     | 300      | -        | -        |

Fig. 2. The architecture of MMIM. In the training process, we select $k_M$ nearest neighbors for each node in each view. And the positive pairs are composed of the common representation of nodes and their nearest neighbors. Meanwhile, the same number of nodes besides the nearest neighbors are randomly selected. And the negative pairs consist of the common representation of nodes and these random nodes. Finally, a discriminator is used to distinguish these pairs.

In our model, we assume that if two samples $x$ and $x'$ are close in any view, their corresponding representations $z$ and $z'$ should also be close in the common latent view. Based on the assumption, a multi-view information maximization module (MMIM) is devised to boost the similarity of inter-neighbour representations. The mutual information maximization module is shown in Fig. 2.

Since larger mutual information denotes the representations are more similar, the mutual information is expected to be as large as possible, and the objective of MMIM can be described as 

$$\max \{I(X, Z')\}$$

According to (2) and (12), the loss function of MMIM $L_{mim}$ can be written as 

$$L_{mim} = -KL(p(z'|x)p(x)||p(z)p(x))$$

However, KL divergence is unbounded. Therefore, we use JS divergence instead of KL divergence in mutual information and (13) can be converted to

$$L_{mim} = -JS(p(z'|x)p(x)||p(z)p(x))$$

According to [36], the variational estimation of JS divergence between two distributions $p(x)$ and $q(x)$ is defined as 

$$JS(p(x)||q(x)) = E_{x \sim p(x)}[\log \rho(T(x))] + E_{x \sim q(x)}[\log(1 - \rho(T(x)))]$$

where $T(x) = \log \frac{2p(x)}{p(x)+q(x)}$ [36]. In our loss function, $p(z'|x)p(x)$ and $p(z)p(x)$ are used to replace $p(x)$ and $q(x)$. Hence, substituting (15) into (14) yields

$$L_{mim} = -E_{(x, z') \sim p(z')p(x)}[\log \rho(T(x, z'))]$$

Here, negative sample estimation [31] is used to solve the problem in (16). Positive and negative sample pairs are generated by the latent representations. Then a discriminator is used to distinguish the negative sample pairs and positive sample pairs to estimate the distribution of positive samples. In (16), $\rho(T(x, z'))$ is a discriminator, and the representation of sample $x$ and its nearest neighbors $x'$ compose positive pairs. The negative pairs are composed of the representation of $x$ and random representations outside the nearest neighbors. For each sample, $k_M$ nearest neighbors are selected to compose positive pairs in all views. For data without a graph, we use $k$-NN algorithm to find the nearest neighbors. And for data with attributed graph, we use a modified Shared Nearest Neighbor (SNN) [37] similarity to find the nearest neighbors on the attributed graph. The modified SNN similarity can be written as

$$\text{sim}(i, j) = \begin{cases} 0, & \text{No edge between } v_i \text{ and } v_j, \\ |N(i) \cap N(j)|, & v_i \text{ and } v_j \text{ are adjacent}. \end{cases}$$

where $N(*)$ denotes the neighboring nodes of $v_*$. After getting the similarity of all other points to $v_*$, the $k_M$ most similar nodes are selected as the nearest neighbors of $v_*$. Thus the total objective function of M-GAE module is defined as:

$$L_M = L_{rec} + \lambda_2 L_{mim}$$

where $\lambda_2$ is a hyperparameter that balances these two loss functions.

IV. EXPERIMENTS

A. Experimental Settings

1) Datasets: In order to fully evaluate the effectiveness of the proposed algorithm, we conduct experiments on three types of multi-view data: (a) Multi-view data without predefined attribute graph, including 3Source,1 BBC,2 100Leaves,3 and Cub;4 (b) Multi-view data with common attribute graph, including Cora, Citeseer and Pubmed;5 (c) Single-view data with multiply attribute graphs, including DBLP,6 IMDB,7 and ACM.8 For convenience, these datasets are summarized in Table I, II and III.

2) Evaluation Metrics: For a comprehensive investigation, we evaluate the performance using five statistical metrics: Accuracy (ACC), Normalized Mutual Information (NMI), Adjusted Mutual Information (AMI), Adjusted Rand Index (ARI) and F1 measure (F1). Generally, the higher values of these five measures mean better clustering quality.

\[ \text{[Online]. Available: http://mlg.ucd.ie/datasets/3sources.html} \]
\[ \text{[Online]. Available: http://mlg.ucd.ie/datasets/segment.html} \]
\[ \text{[Online]. Available: https://archive.ics.uci.edu/ml/datasets/One-hundred-plant+species+leaves+data-set} \]
\[ \text{[Online]. Available: http://www.vision.caltech.edu/visipedia/CUB-200. html} \]
\[ \text{[Online]. Available: https://lqns.soe.ucsc.edu/data} \]
\[ \text{[Online]. Available: https://dlb.uni-trier.de/} \]
\[ \text{[Online]. Available: https://www.imdb.com/} \]
\[ \text{[Online]. Available: http://dl.acm.org} \]
3) Comparison Algorithms: We compare the proposed CMGEC with some single-view clustering methods and several state-of-the-art multi-view clustering.

**Single-view clustering methods:** K-means++ (KM++) [38], graph autoencoder (GAE) [39], deep attentional embedding graph clustering (DAEGC) [18], and structural deep clustering network (SDCN) [40]. For the single-view clustering methods, we report their results of the most informative view (achieves the best clustering performance).

**Multi-view clustering methods:** Partition level multi-view subspace clustering (PMSC) [41], multiview consensus graph clustering (MCGC) [42], multiview graph learning (MVGL) [28], graph-based multi-view clustering(GMC) [23], autoencoder in autoencoder networks(AE²-NET) [43], reciprocal multi-layer subspace learning(RMSL) [25], and One2Multi graph autoencoder clustering framework(O2MAC) [21]. And we perform all algorithms 10 times and report the average results with the standard deviation.

4) Implementation Details: In our experiments, we set $m = 10$, $\lambda_1 = 0.01$, and $\lambda_2 = 0.001$. For multi-view data without predefined graphs, we use $k$-NN algorithm to construct initial graphs. For multi-view data with a common attribute graph, the common graph is copied $V$ times and paired with the data from $V$ views to be fed into GFN and M-GAE. For single-view data with multiply attribute graphs, the single-view data is copied $V$ times and paired with the multiply attribute graphs from $V$ views to be fed into GFN and M-GAE. For each node, $k_M = 3$ nearest neighbors are selected to compose positive pairs. Note that our model is not sensitive to the $k$ of initial $k$-NN graphs in a larger range. We set $k_G = 10$ for all datasets. K-means++ [38] is utilized to obtain the cluster results according to the learned common representation. All the experiments are conducted using the released code on an Ubuntu-18.04 OS with an NVIDIA RTX 3090 GPU. Some methods cannot perform on all types of data, and we only test them on partial datasets.

### B. Comparison of Clustering Performance

**Numerical Results Comparison:** The results of the proposed CMGEC compared with the single-view and multi-view methods on three types of public datasets are shown in Table IV–IX, respectively. The mean and variance of five used metrics are given, with the top value is highlighted in red font and the second-best in blue. Specifically, the performance comparisons of the single-view algorithms with our CMGEC are given in Table IV to VI. From the results in these tables, we reach the following observations: (a) Generally speaking, compared with single-view baselines, the proposed CMGEC achieves better results on all datasets with most metrics, which shows the effectiveness of combining multi-view features for clustering; (b) GCN-based methods outperform the baseline KM++ in most cases, indicating that GCN can learn cluster-friendly embedding.

Moreover, the performance comparisons with multi-view algorithms are given in Table VII to IX. The following observations can be made from the results: (a) Overall, CMGEC achieves very competitive and stable performance compared to almost all multi-view baselines. In many cases, the improvements are very significant. Taking the datasets BBC and 100Leaves for example, the ACC improvements of CMGEC over the second-best baseline are about 3.72% and 9.18%, respectively. The results demonstrate that the common representations learned by our proposed method are effective; (b) CMGEC consistently outperforms subspace-based methods PMSC and RMSL, indicating the effectiveness of learning node representations with the graph structure, because it can extract more inter-node information than just using node features. (c) Although for single-view data with multiple attribute maps, the performance of O2MAC is comparable to our approach. However, it is not easy to apply O2MAC directly to other types of multi-view data; (d) The performance of graph-based shallow methods is generally worse than GCN-based methods, confirming that it is useful to combine node characteristics with adjacent information.

2) **Visualization of the Clustering Results:** In order to show the superiority of the representation obtained by our method, t-SNE [44] is used to visualize the embedded feature space of different methods. And the visualizations on BBC and 3sources are given in Fig. 3. From left to right, they are the space of raw data (best view), the results of GAE (best view), DAEGC (best view), RMSL, and our CMGEC, respectively. From Fig. 3, we can see that the representations obtained by our model are superior than that obtained by other algorithms which have clearer distribution structure.

### C. Parameter Analysis

To better illustrate the stability of the proposed CMGEC, we perform experiments to analysis the sensitivity of the proposed method to the following parameters.

1) **The Parameter Sensitivity of $k_G$ in Predefined Graph:** The number of the nearest neighbors $k_G$ is an important parameter in the construction of the $k$-NN graph for data without predefined graph and has a great impact on the performance of most graph-based algorithms. To examine the effect of $k_G$, we design a $k_G$-sensitivity experiment on the BBC and 100Leaves datasets. It can be seen from Fig 4 that our model is insensitive with $k_G \in \{5, 20\}$ compared with GMC and DAEGC. It proves that our

### TABLE II

**DATASETS STATISTICS OF MULTI-VIEW DATA WITH COMMON ATTRIBUTE GRAPH**

| Datasets | Classes | Nodes | Edges | Attribute1 | Attribute2 |
|----------|---------|-------|-------|------------|------------|
| Cora     | 7       | 2708  | 5429  | 1433       | 2708       |
| Citeese  | 6       | 3327  | 4732  | 3703       | 3327       |
| Pubmed   | 3       | 19717 | 44338 | 19717      | 500        |

### TABLE III

**DATASETS STATISTICS OF SINGLE-VIEW DATA WITH MULTIPLY ATTRIBUTE GRAPHS**

| Datasets | Classes | Nodes | Dimension | Edge1 | Edge2 | Edge3 |
|----------|---------|-------|-----------|-------|-------|-------|
| DBLP     | 4       | 4057  | 334       | 11113 | 5000495 | 6776335 |
| IMDB     | 3       | 4780  | 1232      | 98010 | 21018  | -     |
| ACM      | 3       | 3025  | 1830      | 29281 | 2210761 | -     |

WANG et al.: CONSISTENT MULTIPLE GRAPH EMBEDDING FOR MULTI-VIEW CLUSTERING
### TABLE IV
**Performance Comparisons Between Single View Methods and CMGEC on Multi-View Data Without Predefined Attribute Graph**

| Datasets | Methods | ACC    | NMI    | ARI     | AMI     | F1     |
|----------|---------|--------|--------|---------|---------|--------|
| 3Sources | KM++   | 0.5300 | 0.3067 | 0.3420 | 0.1035 | 0.2880 | 0.1300 |
|          | GAE    | 0.6765 | 0.5155 | 0.5756 | 0.0476 | 0.4533 | 0.0446 |
|          | DAECG  | 0.7160 | 0.6000 | 0.6066 | 0.0000 | 0.6208 | 0.0000 |
|          | SDCN   | 0.6252 | 0.0028 | 0.4230 | 0.0088 | 0.4225 | 0.0143 |
|          | CMGEC  | 0.7653 | 0.0307 | 0.6694 | 0.0143 | 0.6049 | 0.0446 |
| BBC      | KM++   | 0.5201 | 0.0673 | 0.3516 | 0.0732 | 0.2180 | 0.1006 |
|          | GAE    | 0.6397 | 0.0666 | 0.5263 | 0.0291 | 0.4720 | 0.0437 |
|          | DAECG  | 0.6746 | 0.0000 | 0.5278 | 0.0000 | 0.4661 | 0.0000 |
|          | SDCN   | 0.7116 | 0.0044 | 0.5713 | 0.0020 | 0.5082 | 0.0067 |
|          | CMGEC  | 0.8737 | 0.0061 | 0.7144 | 0.0119 | 0.7392 | 0.0113 |

### TABLE V
**Clustering Results Between Single View Methods and CMGEC on Multi-View Data With Common Attribute Graph**

| Datasets | Methods | ACC    | NMI    | ARI     | AMI     | F1     |
|----------|---------|--------|--------|---------|---------|--------|
| Cora     | KM++   | 0.3311 | 0.0322 | 0.1302 | 0.0334 | 0.5097 | 0.0194 |
|          | GAE    | 0.5010 | 0.0386 | 0.3971 | 0.0259 | 0.2933 | 0.0243 |
|          | DAECG  | 0.6969 | 0.0002 | 0.5341 | 0.0004 | 0.4690 | 0.0001 |
|          | SDCN   | 0.6024 | 0.0043 | 0.5004 | 0.0100 | 0.3902 | 0.0029 |
|          | CMGEC  | 0.7068 | 0.0034 | 0.4851 | 0.0184 | 0.4172 | 0.0204 |
| Citeseer | KM++   | 0.4755 | 0.0584 | 0.2338 | 0.0457 | 0.2002 | 0.0416 |
|          | GAE    | 0.3802 | 0.0167 | 0.1746 | 0.0179 | 0.1613 | 0.0213 |
|          | DAECG  | 0.6595 | 0.0031 | 0.4168 | 0.0000 | 0.4152 | 0.0000 |
|          | SDCN   | 0.6596 | 0.0031 | 0.3871 | 0.0032 | 0.4017 | 0.0043 |
|          | CMGEC  | 0.6765 | 0.0052 | 0.3666 | 0.0361 | 0.4072 | 0.0361 |
| Pubmed   | KM++   | 0.5989 | 0.0009 | 0.3114 | 0.0031 | 0.2814 | 0.0014 |
|          | GAE    | 0.6324 | 0.0167 | 0.2497 | 0.0025 | 0.2460 | 0.0268 |
|          | DAECG  | 0.6712 | 0.0000 | 0.2663 | 0.0001 | 0.2782 | 0.0001 |
|          | SDCN   | 0.6578 | 0.0042 | 0.2947 | 0.0054 | 0.2546 | 0.0059 |
|          | CMGEC  | 0.7055 | 0.0087 | 0.3428 | 0.0043 | 0.3345 | 0.0050 |

### TABLE VI
**Clustering Results Between Single View Methods and CMGEC on Single-View Data With Multiply Attribute Graphs**

| Datasets | Methods | ACC    | NMI    | ARI     | AMI     | F1     |
|----------|---------|--------|--------|---------|---------|--------|
| DBLP     | KM++   | 0.3664 | 0.1006 | 0.1153 | 0.0049 | 0.0671 | 0.0080 |
|          | GAE    | 0.5558 | 0.0199 | 0.3072 | 0.0073 | 0.2577 | 0.0061 |
|          | DAECG  | 0.8733 | 0.0000 | 0.6742 | 0.0000 | 0.7014 | 0.0000 |
|          | SDCN   | 0.6497 | 0.0039 | 0.2977 | 0.0018 | 0.3095 | 0.0033 |
|          | CMGEC  | 0.9103 | 0.0039 | 0.7237 | 0.0021 | 0.7859 | 0.0062 |
| IMDB     | KM++   | 0.3154 | 0.0034 | 0.0119 | 0.0048 | 0.0028 | 0.0021 |
|          | GAE    | 0.4298 | 0.0134 | 0.0402 | 0.0031 | 0.0403 | 0.0019 |
|          | DAECG  | 0.3863 | 0.0013 | 0.0555 | 0.0004 | 0.0309 | 0.0001 |
|          | SDCN   | 0.4047 | 0.0030 | 0.0099 | 0.0009 | 0.0109 | 0.0011 |
|          | CMGEC  | 0.4844 | 0.0123 | 0.0514 | 0.0091 | 0.0469 | 0.0077 |
| ACM      | KM++   | 0.6753 | 0.1113 | 0.3253 | 0.0047 | 0.3077 | 0.0106 |
|          | GAE    | 0.6992 | 0.0161 | 0.4771 | 0.0083 | 0.4377 | 0.0070 |
|          | DAECG  | 0.8909 | 0.0000 | 0.6430 | 0.0000 | 0.7046 | 0.0000 |
|          | SDCN   | 0.8631 | 0.0052 | 0.5783 | 0.0088 | 0.5837 | 0.0110 |
|          | CMGEC  | 0.9898 | 0.0073 | 0.6012 | 0.0036 | 0.7232 | 0.0106 |

Method can learn multi-view structural information even there are less neighbor information or some spurious connections. However, larger \( k_G \) can lead to more edges in the graph, which slows down the speed of the graph convolution. Thus, we set \( k_G \) to 10 for all datasets in our experiments.

2) The Parameter Effect of \( \lambda_1 \) and \( \lambda_2 \):
In our CMGEC model, there are two hyperparameters \( \lambda_1 \) and \( \lambda_2 \) that need to be set properly. In our experiments, we tune \( \lambda_1 \) and \( \lambda_2 \) from \{0.0001, 0.001, 0.01, 1, 0.1, 1\} and \{0.0001, 0.001, 0.01, 0.1, 1\}, respectively. Fig. 5 shows the results of our method using different parameters (taking BBC as an example). Here, we vary a parameter at a time while keeping another fixed. From Fig 5, it can be seen that our method performs stably over a wide range of hyperparameter values.
3) The Parameter Analysis of $k_M$: In MMIM, for each node, $k_M$ nearest neighbors are selected to compose positive pairs. In the following, we conduct experiments to show the effect of this parameter on the clustering performance. Fig 6 presents the ACC and NMI of CMGEC by varying $k_M$ from 1 to 15. We can observe that the metrics first increases to a high value and generally maintains it up to slight variation with the increasing of $k_M$. CMGEC demonstrates stable performance across a wide range of $k_M$. For time-consuming reasons, we set $k_M$ to 3 in our experiments.

4) The Sensitivity of the Dimension of the Common Representation ($m$): We vary the dimension of the learned representation from 5 to 40 and the results is given in Table X. It can be observed that: when the dimension of representation changes from 5 to 10, the clustering performance improves significantly; however, when the dimension continue to increase, the clustering performance fluctuates, but the overall performance is still good. On the other hand, the time consumption of the algorithm increases with the dimensionality. Therefore, we set $m = 10$ in our experiment.
D. Ablation Study

In this section, the impact of each part on our CMGE model are analyzed in detail. Specifically, we divided the ablation experiments into two parts according to how the clustering results are obtained as follows:

1) Clustering With Consensus Graph and Predefined Graph (Only With GFN): Generally, the clustering results can be obtained from the representation or the graph segmentation [45]. In order to show the effectiveness of GFN module, we conduct comparison experiments on four datasets. The clustering results are given in Fig. 7, where PGS denotes the clustering results obtained from the segmentation of the informative predefined graph, and CMGE-cgg denotes the clustering results obtained by the segmentation of the consensus graph $A^*$ learned by GFN. From Fig. 7, it can be seen that CMGE-cgg performs better than PGS in all datasets, showing the effectiveness of learning consensus graph using GFN. Moreover, CMGE outperforms PGS and CMGE-cgg in all datasets, indicating that graph embedding can help to learn a suitable representation for clustering compared to consensus graph segmentation clustering.

2) Clustering Without MMIM or GFN: To further investigate the effectiveness of diverse components of our model, we perform the following experiments to isolate the effect of GFN and MMIM. The clustering results are shown in Table XI, where
CMGEC-MG (first row) means using M-GAE with $L_{rec}$ to obtain the common representation. CMGEC-G (second row) denotes M-GAE is trained using the whole loss $L_M$, but the graph of the most informative view is used as the consensus graph, CMGEC-M (third row) means M-GAE is trained using $L_{rec}$ and the consensus graph provided by GFN, and CMGEC (last row) means using all components to obtain the clustering results. From Table XI, we observe that: (a) It can be seen that each variant of our method has relatively high ACC, NMI, and ARI, and the best performance can be achieved when using whole CMGEC, which demonstrates that each part of the proposed model is significant for clustering task. (b) CMGEC-G generally achieves better cluster results than CMGEC-MG which suggests that mutual information contributes to the learning of more discriminative common representations; (c) We can clearly notice that CMGEC-M significantly outperforms CMGEC-MG and CMGEC-G. It shows that learning a unified graph is essential to the learning of a suitable common representation and that using any view of the graph as the consensus graph is prejudiced.

V. CONCLUSION

In this paper, we propose a Consistent Multiple Graph Embedding Clustering framework (CMGEC), which is mainly composed of Multiple Graph Auto-Encoder (M-GAE), Multi-view Mutual Information Maximization module (MMIM) and Graph Fusion Network (GFN). Specifically, M-GAE is devised to learn a common representation using a multi-graph attention fusion encoder and reconstruct multi-view graphs by view-specific decoders. By introducing a multi-graph attention fusion layer, the common representation can adaptively integrate complementary information from multiple views. In order to maintain the similarity of the neighboring characteristic, MMIM is introduced to make similar instances still similar to each other in the common space. Moreover, we design a GFN to explore complex relationships among different views and learn a consensus graph needed in M-GAE. And the rank constraint on its Laplacian matrix is further utilized to train the GFN to improve the separability of the consensus graph. By jointly training these models, a view consistent representation can be learned for clustering. Experiments on three types of multi-view datasets verify the advantage of our proposed method compared with state-of-the-art methods.

REFERENCES

[1] X. Gao, T. Mu, J. Y. Goutlermas, and M. Wang, “Topic driven multimodal similarity learning with multi-view voted convolutional features,” Pattern Recognit., vol. 75, pp. 223–234, 2018.
[2] Y. Chen, X. Xiao, and Y. Zhou, “Jointly learning kernel representation tensor and affinity matrix for multi-view clustering,” IEEE Trans. Multimedia, vol. 22, no. 8, pp. 1985–1997, Aug. 2020.
[3] G. Chao, S. Sun, and J. Bi, “A survey on multiview clustering,” IEEE Trans. Artif. Intell., vol. 2, no. 2, pp. 146–168, Apr. 2021.
[4] G. Chao et al., “Multi-view cluster analysis with incomplete data to understand treatment effects,” Inf. Sci., vol. 494, pp. 278–293, 2019.
[5] H. Riemenschneider, A. Bódiz-Szomorú, J. Weissenberg, and L. V. Gool, “Learning where to classify in multi-view semantic segmentation,” in Proc. Eur. Conf. Comput. Vis., 2014, vol. 8693, pp. 516–532.
[6] M. Yin, J. Gao, S. Xie, and Y. Guo, “Multiview subspace clustering via tensorial t-product representation,” IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 3, pp. 851–864, Mar. 2019.
[7] Y. Wang, X. Lin, L. Wu, W. Zhang, Q. Zhang, and X. Huang, “Robust subspace clustering for multi-view data by exploiting correlation consensus,” IEEE Trans. Image Process., vol. 24, no. 11, pp. 3939–3949, Nov. 2015.
[8] Y. Zheng, “Methodologies for cross-domain data fusion: An overview,” IEEE Trans. Big Data, vol. 1, no. 1, pp. 16–34, Mar. 2015.
[9] F. Nie, J. Li, and X. Li, “Self-weighted multiview clustering with multiple graphs,” in Proc. Int. Joint Conf. Artif. Intell., 2017, pp. 2564–2570.
[10] Z. Kang et al., “Multi-graph fusion for multi-view spectral clustering,” Knowl. Based Syst., vol. 189, 2020, Art. no. 105102.
[11] C. Zhang et al., “Generalized latent multi-view subspace clustering,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 1, pp. 86–99, Jan. 2020.
[12] S. Wang et al., “Multi-view clustering via late fusion alignment maximization,” in Proc. Int. Joint Conf. Artif. Intell., 2019, pp. 3778–3784.
[13] X. Liu et al., “Late fusion incomplete multi-view clustering,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 41, no. 10, pp. 2410–2423, Oct. 2019.

---

TABLE XI

| Method | M-GAE | MMIM | GFN | ACC | NMI | ARI | ACC | NMI | ARI | ACC | NMI | ARI |
|--------|-------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| CMGEC-MG | ✓    | ✓   | ✓  | 0.6499 | 0.6360 | 0.7373 | 0.7899 | 0.7497 | 0.6091 | 0.6597 | 0.6401 | 0.6603 | 0.7149 |
| CMGEC-G | ✓    | ✓   | ✓  | 0.5684 | 0.6364 | 0.7373 | 0.7899 | 0.7497 | 0.6091 | 0.6597 | 0.6401 | 0.6603 | 0.7149 |
| CMGEC-M | ✓    | ✓   | ✓  | 0.7549 | 0.6694 | 0.6049 | 0.6467 | 0.7951 | 0.7117 | 0.8779 | 0.7144 | 0.7392 | 0.9899 |
| CMGEC | ✓    | ✓   | ✓  | 0.7549 | 0.6694 | 0.6049 | 0.6467 | 0.7951 | 0.7117 | 0.8779 | 0.7144 | 0.7392 | 0.9899 |

Different methods use modules identified by ✓. Best results are highlighted in Bold.
[14] K. Zhan, C. Niu, C. Chen, F. Nie, C. Zhang, and Y. Yang, “Graph structure fusion for multiview clustering,” IEEE Trans. Knowl. Data Eng., vol. 31, no. 10, pp. 1984–1993, Oct. 2019.

[15] J. Wen et al., “Adaptive graph completion based incomplete multi-view clustering,” IEEE Trans. Multimedia, vol. 23, pp. 2493–2504, 2021.

[16] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in Proc. Int. Conf. Learn. Represent., 2017, pp. 1–14.

[17] M. S. Schlichtkrull et al., “Modeling relational data with graph convolutional networks,” in Proc. Eur. Semantic Web Conf., 2020, vol. 10843, pp. 593–607.

[18] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang, and C. Zhang, “Attributed graph clustering: A deep attentional embedding approach,” in Proc. Int. Joint Conf. Artif. Intell., 2019, pp. 3670–3676.

[19] Z. Wang, L. Zheng, Y. Li, and S. Wang, “Linkage based face clustering via graph convolution network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 1117–1125.

[20] Z. Tao, H. Liu, J. Li, Z. Wang, and Y. Fu, “Adversarial graph embedding for ensemble clustering,” in Proc. Int. Joint Conf. Artif. Intell., 2019, pp. 3562–3568.

[21] S. Fan, X. Wang, C. Shi, E. Lu, K. Lin, and B. Wang, “One2multi graph autoencoder for multi-view graph clustering,” in Proc. World Wide Web Conf., 2020, pp. 3070–3076.

[22] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, “Graph transformer networks,” in Proc. Adv. Neural Inform. Process. Syst., 2019, pp. 11900–11907.

[23] H. Wang, Y. Yang, and B. Liu, “GMC: Graph-based multi-view clustering,” IEEE Trans. Knowl. Data Eng., vol. 32, no. 6, pp. 1116–1129, Jun. 2020.

[24] M. Sun et al., “Projective multiple kernel subspace clustering,” IEEE Trans. Multimedia, early access, Jun. 2021.

[25] R. Li, C. Zhang, H. Fu, X. Peng, J. T. Zhou, and Q. Hu, “Reciprocal multi-layer subspace learning for multi-view clustering,” in Proc. Int. Conf. Comput. Vis., 2019, pp. 8171–8179.

[26] N. F. Chikhi, “Multi-view clustering via spectral partitioning and local refinement,” Inf. Process. Manag., vol. 52, no. 4, pp. 618–627, 2016.

[27] J. W. Son, J. Jeon, A. Lee, and S. Kim, “Spectral clustering with brainstorming process for multi-view data,” in Proc. AAAI Conf. Artif. Intell., 2017, pp. 2548–2554.

[28] K. Zhan, C. Zhang, J. Guan, and J. Wang, “Graph learning for multi-view clustering,” IEEE Trans. Cybern., vol. 48, no. 10, pp. 2887–2895, Oct. 2018.

[29] M. I. Belghazi et al., “Mutual information neural estimation,” in Proc. Int. Conf. Mach. Learn., 2018, vol. 80, pp. 530–539.

[30] R. Gomes, A. Krause, and P. Perona, “Discriminative clustering by regularized information maximization,” in Proc. Adv. Neural Inform. Process. Syst., 2010, pp. 775–783.

[31] R. D. Hjelm et al., “Learning deep representations by mutual information estimation and maximization,” in Proc. Int. Conf. Learn. Represent., 2019, pp. 1–24.

[32] P. Bachman, R. D. Hjelm, and W. Buchwalter, “Learning representations by maximizing mutual information across views,” in Proc. Adv. Neural Inform. Process. Syst., 2019, pp. 15509–15519.

[33] Y. Mao, X. Yan, Q. Guo, and Y. Ye, “Deep mutual information maximin for cross-modal clustering,” in Proc. AAAI Conf. Artif. Intell., 2021, pp. 8893–8901.

[34] C. Li and S. Lin, “Social flocks: Simulating crowds to discover the connection between spatial-temporal movements of people and social structure,” IEEE Trans. Comput. Soc. Syst., vol. 5, no. 1, pp. 33–45, Mar. 2018.

[35] J. Wen, N. Han, X. Fang, L. Fei, K. Yan, and S. Zhan, “Low-rank preserving projection via graph regularized reconstruction,” IEEE Cybern., vol. 49, no. 4, pp. 1279–1291, Apr. 2019.

[36] W. Jiang, W. Liu, and F. Chung, “Knowledge transfer for spectral clustering,” Pattern Recognit., vol. 81, pp. 484–496, 2018.

[37] R. A. Jarvis and E. A. Patrick, “Clustering using a similarity measure based on shared near neighbors,” IEEE Trans. Comput., vol. 100, no. 11, pp. 1025–1034, Nov. 1973.

[38] D. Sculley, “Web-scale k-means clustering,” in Proc. 19th Int. Conf. World Wide Web Conf., M. Rappa, P. Jones, I. Freire, and S. Chakrabarti, Eds., 2010, pp. 1177–1178.

[39] T. N. Kipf and M. Welling, “Variational graph auto-encoders,” in Proc. NIPS Workshop Bayesian Deep Learn., 2016, pp. 1–3, arXiv:1611.07308.

[40] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, and P. Cui, “Structural deep clustering network,” in Proc. World Wide Web Conf., 2020, pp. 1400–1410.

[41] Z. Kang et al., “Partition level multiview subspace clustering,” Neural Netw., vol. 122, pp. 279–288, 2020.