Deep Fusion Clustering Network with Reliable Structure Preservation

Lei Gong, Wenxuan Tu, Yue Liu, Sihang Zhou, Tiantian Xu, Xinwang Liu, *Senior Member, IEEE*

***Abstract*—Deep clustering, which can elegantly exploit data representation to seek a partition of the samples, has attracted intensive attention. Recently, combining auto-encoder with graph neural networks (GNNs) has accomplished excellent performance by introducing structural information implied among data in clustering tasks. However, we observe that there are some limitations of most existing works: 1) in practical graph datasets, there exist some noisy or inaccurate connections among nodes, which would confuse the network learning and cause biased representations, thus learning to unsatisfied clustering performance; 2) lacking dynamic information fusion module to carefully combine and refine the node attributes and the graph structural information to learn more consistent representations; 3) failing to exploit the two separated views information for generating a more robust target distribution. To solve these problems, we propose a novel method termed deep fusion clustering network with reliable structure preservation (DFCN-RSP). Specifically, the random walk mechanism is introduced to boost the reliability of the original graph structure by measuring localized structure similarities among nodes. It can simultaneously filter out noisy connections and supplement reliable connections in the original graph. Moreover, we provide a transformer-based graph auto-encoder (TGAE) which can utilize a self-attention mechanism with the localized structure similarity information to fine-tune the fused topology structure among nodes layer-by-layer. Further, we provide a dynamic cross-modality fusion strategy to combine the representations learned from both TGAE and auto-encoder. Also, we design a triplet self-supervision strategy as well as a target distribution generation measure to explore the cross-modality information. The experimental results on five public benchmark datasets reflect that DFCN-RSP is more competitive than the state-of-the-art deep clustering algorithms.**

***Index Terms*—Deep Clustering, Graph Neural Network, Graph Transformer, Graph Structure Learning, Latent Feature Fusion.**

I. Introduction

DEEP clustering, which achieves great success on the tasks of dividing data into several disjoint groups without manual labels by neural networks, has been an indispensable part of the clustering family. Due to the powerful capacity of deep learning techniques, deep clustering methods increasingly become a natural way to solve the data annotation issue to make full use of large amounts of unlabeled data. Moreover, the corresponding methods have achieved great success in many applications including social network analysis [1], anomaly detection [2], and face recognition [3], to name just a few.

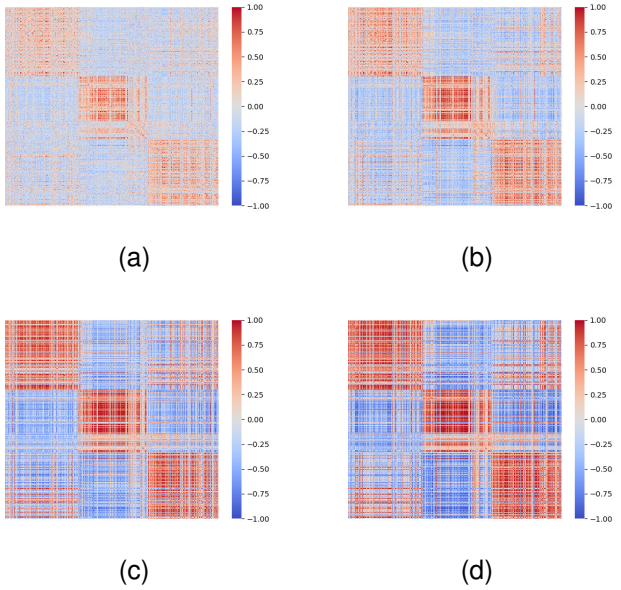

Fig. 1. Latent feature discriminative capability illustration. We first permute the samples to have those belonging to the same cluster located beside each other. Then we calculate the cosine similarities among the node embeddings learned by different methods. The corresponding methods are (a) GAE, (b) GAE with weighted adjacent matrix only, (c) GAE with self-attention architecture, and (d) TGAE on dataset ACM.

Recent efforts have been devoted to deep clustering domain and can be mainly divided into five categories, i.e., spectral clustering-based methods [4]–[6], subspace clustering-based methods [7]–[12], generative adversarial network-based methods [13]–[15], Gaussian mixture model-based methods [16], [17], and self-optimizing-based methods [18], [19]. Our proposed DFCN-RSP belongs to the last one. Specifically, the early methods learn data representations in a low dimensional space by reconstructing attribute features within an auto-encoder architecture [18], [19]. Then, the structure information of data attracts extensive attention and has been utilized to improve the clustering performance. Representative algorithms include GMM [16], DPSC [10], and DAEGC [20]. They attempt to integrate structural information with node attribute features to further improve the discriminability of learned node representations. However, owing to the complex node attributes as well as the non-Euclidean graph structure, most

L. Gong, W. Tu, Y. Liu, and X. Liu are with the School of Computer, National University of Defense Technology, Changsha, 410073, China (e-mail: {glnudt, wenxuantu, yueliu19990731}@163.com, xinwangliu@nudt.edu.cn).

S. Zhou is with the College of Intelligence Science and Technology, National University of Defense Technology, Changsha, 410073, China (e-mail: sihangjoe@gmail.com).

T. Xu is with the Department of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan, China, (email: xtt-ok@163.com).

L. Gong and W. Tu are the first authors with equal contributions.

Corresponding author: X. Liu.

of the existing deep clustering methods cannot be used to analyze the graph-structured data directly. To handle this issue, in recent years, graph convolutional networks (GCNs) [21] have made great development and gained much attention from researchers. GCNs aim to aggregate the neighbors information for graph representation learning and have shown great success in graph-oriented clustering tasks. For instance, the work in [20] proposes a deep attention embedded graph clustering (DAEGC) algorithm that transforms the node attributes and topological structure into a compact embedding space and then reconstructs adjacent matrix by a self-optimizing embedding method. Inspired by this work, the adversarially regularized graph auto-encoder (ARGA) [22] is developed to learn node representations under the guidance of an adversarial learning mechanism. In addition, multi-view graph representation learning (MVGRL) [23] conducts cross-view contrastive learning using two decoupled GCN-based encoders to further improve the quality of node representations for clustering performance enhancement. Furthermore, the structural deep clustering network (SDCN) [24] designs a dual self-supervised strategy and an information delivery operator to integrate the auto-encoder and GCNs into a unified framework to improve the performance of clustering.

Although the aforementioned methods have obtained promising performance by integrating information from node attributes and graph structure, there still exist some limitations. **First**, most graph-oriented methods seldom consider refining the raw graph structure, however, in practical graph datasets, we observe that there exist some noisy or inaccurate connections among nodes, where two nodes with different ground-truth labels keep linkage relations. These unreliable relationships would confuse the network learning and cause biased representations, thus learning to unsatisfied clustering performance. **Second**, current algorithms lack a dynamic cross-modality fusion mechanism to carefully integrate the node attributes and graph structural information for learning consistent representations. In existing algorithms, the resultant representations from two-source information are usually directly concatenated or aligned, leading to insufficient information negotiation. **Third**, in most current literature, algorithms fail to sufficiently utilize the two-source information to generate the target distribution for guiding the network learning. This prevents the learned graph representations from being more precise and comprehensive. Consequently, the negotiation between the node attribute and graph structure is disconnected, thus leading to sub-optimal clustering performance.

Based on the observations, we propose a novel method termed deep fusion clustering network with reliable structure preservation (DFCN-RSP), which is an improved version of our deep fusion clustering network (DFCN) [25], to address the above issues. The key points in our solution are three-fold: 1) we refine the original graph via the random walk mechanism, which can filter out noisy connections and supplement reliable connections simultaneously; 2) we propose a transformer-based graph auto-encoder (TGAE), which utilizes a more reliable and informative topology structure to capture accurate linkage relationships in each layer for better representation learning; 3) we design a dynamic cross-modality fusion strategy, where the structural information and node attributes are integrated to achieve representation consistency for conducting a robust target distribution. To be specific, the random walk mechanism is first introduced to boost the reliability of the original graph structure by measuring localized structural similarities among nodes. In this way, the noisy connections in the original graph can be filtered and some reliable connections can be supplemented, simultaneously. Then, inspired by the transformer mechanism [26], we provide a transformer-based graph auto-encoder (TGAE). We integrate the node attributes and structural information to generate a fused topology, which can accomplish the information propagation and aggregation among high-order nodes in each layer to improve the representation learning and clustering performance. In Fig. 1, we visually illustrate the learned structural embeddings of the graph auto-encoder (GAE) in DFCN [25] and our improved transformer-based graph auto-encoder (TGAE) on dataset ACM. We observe that our proposed TGAE could learn more discriminative node representations for clustering. In addition, we develop a dynamic cross-modality fusion strategy to integrate the two-source information from graph structure and node attributes for consistent representation learning. After that, to generate a more robust target distribution, we estimate the similarities between nodes and pre-calculated cluster centers with students' t-distribution in the latent embedding space. Finally, we design a triplet self-supervision framework to guide the learning processes of the fusion part, AE, and TGAE for network optimization. Our contributions to this work are summarized below:

- We propose a novel method termed deep fusion clustering network with reliable structure preservation (DFCN-RSP), which is the first algorithm to provide a weighted adjacent matrix according to localized structural similarity and further improve the efficiency of GCNs with a more reliable and informative topology structure to capture both the first-order and the high-order relationships among nodes in each layer.
- To refine the original adjacent matrix and boost the clustering performance, we explore the weighted adjacent matrix generation by the random walk mechanism to simultaneously filter out noisy edges and supplement reliable edges in the original graph. Then the first-order and the high-order relationships among nodes extracted by the TGAE are used to fine-tune the global topology layer-by-layer for seeking informative features.
- We design a dynamic cross-modality fusion module to improve the generalization ability by integrating the representations learned from both AE and TGAE for consistent representations.
- Experimental results on five public benchmark datasets have verified that DFCN-RSP achieves superior performance compared with its counterparts.

II. RELATED WORK

A. Deep Graph Clustering

Graph convolution networks (GCNs) [21] have revealed great performance in deep clustering with their excellent

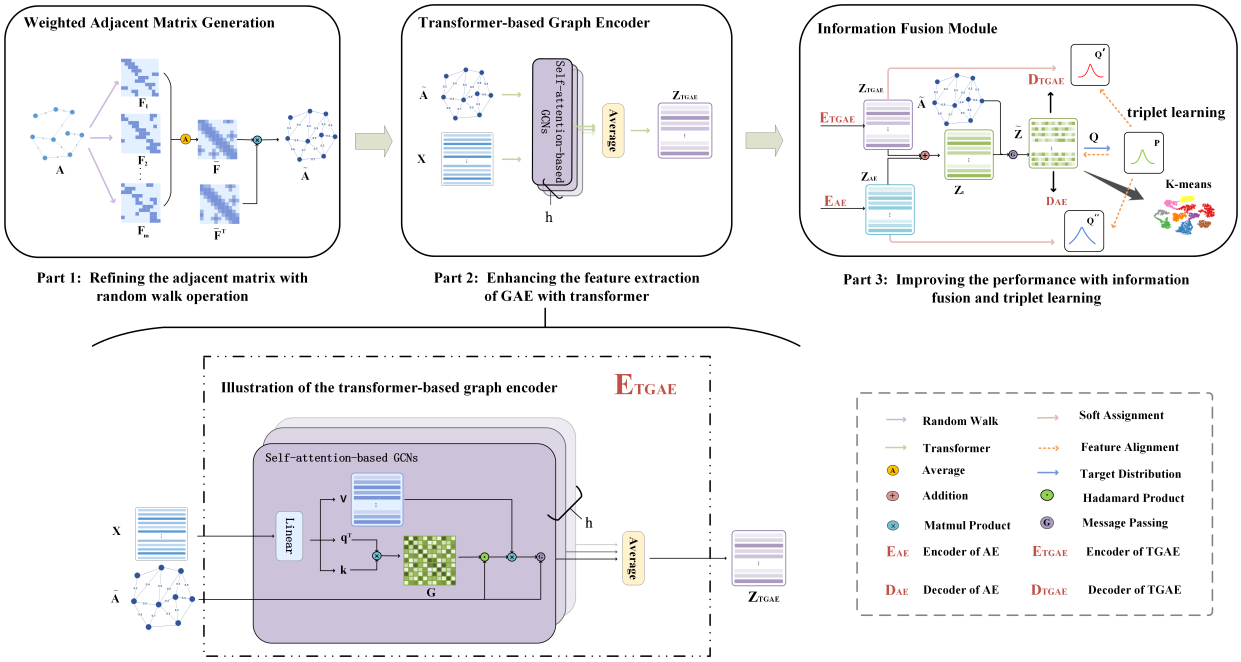

Fig. 2. Structure illustration of DFCN-RSP. The proposed algorithm can be roughly divided into three parts. Part 1: refine the adjacent matrix and generate the weighted adjacent matrix by the random walk mechanism based on the raw similarity graph $\mathbf{A}$, $\mathbf{F}_i$ is the i-th random walk matrix, and $\bar{\mathbf{F}}$ is the average random walk matrix; Part 2: generates two feature embeddings with an auto-encoder(AE) and a transformer enhanced graph auto-encoder(TGAE), respectively; Part 3: fusing the embeddings from AE and TGAE and calculating the loss function for network training.

capability for representation learning on graph-structure data, recently. Variational graph auto-encoder (VGAE) [27] is introduced to integrate the node embedding learning process with graph structure information, and then reconstruct the adjacent matrix in the decoder. Since introducing a self-attention mechanism into GCNs structure, deep attentional embedded graph clustering (DAEGC) [20] can capture the informative neighborhood information for exploring more discriminative node embeddings. Further, some advanced methods such as AGAE [28], ARGA [29], and MinCutPool [30] are proposed to improve the clustering performance by introducing adversarial learning mechanism and graph pooling strategy, respectively. However, most aforementioned methods optimize the network by merely reconstructing adjacent matrices while ignoring the exploitation of the node attributes. To solve this issue, recent advances such as SDCN [24], DFCN [25], and DCRN [31] consider to utilize both the structure and node attributes information for clustering via auto-encoder and graph auto-encoder. However, their clustering performances are still limited due to relying on first-order neighbors and neglecting high-order relationships, which leads to less discriminative representations. Differently, inspired by the transformer mechanism [26], we propose a transformer-based graph auto-encoder that can simultaneously explore the first-order neighbors information and the long dependencies among nodes in each layer, which achieves learning more discriminative representations, thus improving the clustering performance.

B. Graph Structure Learning

Graph structure learning has increasingly gained much attention in recent years [32]–[35]. To improve the quality of the learned node representations, graph machine learning researchers make much effort to generate a more accurate and informative graph structure for representation learning. Specifically, the graph signal processing-based methods [36]–[38] are developed to handle this issue. Moreover, matrix factorization-based methods [39]–[41] are proposed to learn graph structure with graph Laplacian and vertex proximity techniques, respectively. In addition, the two classical random walk-based methods, i.e., node2vec [42] and deep walk [43], exploit random walk to learn and optimize the graph structure. These methods regard the nodes as words and exploit the vertex sequences generated by random walk as the inputs of language models, which are optimized via maximizing the co-occurrence probability of vertices. Since the powerful learning capacity of the graph neural networks (GNNs) [44], [45], GAE [27] employs a GNN-based encoder to extract the semantic features of nodes and rebuilt the adjacent matrix via a simple decoder, i.e., inner product operation. To further improve the quality of graph representations, SDCN [24] and DFCN [25] combine AE and GAE to integrate information from node characteristics and original graph structure simultaneously. Despite their success, most GNN-based methods follow the principle that the original graph structure is accurate. However, this assumption may not hold in practical graph datasets. In contrast, in our method, we reconstruct a graph that can reveal localized structural similarity via k-step random walk. Specifically, similar to deep walk and node2vec, we first obtain the walk sequences for each node by the random walk strategy and then evaluate the similarities of the walk sequences (i.e., the localized structural similarities) among sample-wise pairs. By this means, we can obtain a sample-wise correlation-based

TABLE I
NOTATION SUMMARY

Notations	Meaning
$\mathbf{X} \in \mathbb{R}^{N\times d}$	Attribute matrix
$\hat{\mathbf{X}} \in \mathbb{R}^{N\times d}$	Reconstructed attribute matrix
$\mathbf{A} \in \mathbb{R}^{N\times N}$	Raw adjacent matrix
$\tilde{\mathbf{A}} \in \mathbb{R}^{N\times N}$	Weighted adjacent matrix
$\hat{\mathbf{A}} \in \mathbb{R}^{N\times N}$	Reconstructed adjacent matrix
$\mathbf{G} \in \mathbb{R}^{N\times N}$	Sample correlation-based topology matrix
$\tilde{\mathbf{G}} \in \mathbb{R}^{N\times N}$	Fused topology matrix
$\mathbf{F}_i \in \mathbb{R}^{N\times N}$	i-th random walk matrix
$\bar{\mathbf{F}} \in \mathbb{R}^{N\times N}$	Averaged random walk matrix
$\mathbf{Z}_{AE} \in \mathbb{R}^{N\times d'}$	Latent representations of AE
$\mathbf{Z}_{TGAE} \in \mathbb{R}^{N\times d'}$	Latent representations of TGAE
$\mathbf{Z}_I \in \mathbb{R}^{N\times d'}$	Initial fused embedding
$\tilde{\mathbf{Z}} \in \mathbb{R}^{N\times d'}$	Clustering embedding
$\mathbf{B} \in \mathbb{R}^{N\times K}$	Soft assignment distribution
$\mathbf{P} \in \mathbb{R}^{N\times K}$	Target distribution
$\mathbf{Q}^{(l)} \in \mathbb{R}^{N\times d^{(l)}}$	Query matrix of l-th layer
$\mathbf{K}^{(l)} \in \mathbb{R}^{N\times d^{(l)}}$	Key matrix of l-th layer
$\mathbf{V}^{(l)} \in \mathbb{R}^{N\times d^{(l)}}$	Value matrix of l-th layer

adjacent matrix, which could filter out noisy connections as well as supplement reliable connections in the raw graph, and provide more reliable guidance for clustering.

III. METHOD

In this section, the proposed method termed deep fusion clustering network with reliable structure preservation (DFCN-RSP) will be introduced, which is an improved version of DFCN [25]. This method aims to address the biased and weak guidance caused by the inaccurate graph structure via the random walk technique. As the illustration of Fig. 2, the proposed DFCN-RSP mainly contains three parts, i.e., weighted adjacent matrix refined by the random walk mechanism, transformer-based graph auto-encoder (TGAE), and the dynamic information fusion module. We first summarize the basic notations associated with our proposed method and then introduce the generation procedure of the weighted adjacent matrix, the improved TGAE, and the dynamic information fusion module as below. Note that the details of the triplet self-supervised strategy are similar to DFCN and can be found in corresponding literature [25].

A. Notations

We denote that $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with K clustering centers is an undirected graph, where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ is the nodes set and $\mathcal{E}$ is the edges set, $|\mathcal{V}| = N$. $\mathbf{X} \in \mathbb{R}^{N\times d}$ is the nodes attribute matrix and d is the attribute dimension. $\mathbf{A} = (a_{ij})_{N\times N} \in \mathbb{R}^{N\times N}$ is the raw adjacent matrix, if $a_{ij} = 1$, there is $(v_i, v_j) \in \mathcal{E}$, otherwise, $a_{ij} = 0$. All the notations are summarized in Table I.

B. Weighted Adjacent Matrix Generation

In practical graph datasets, we observe that there exist some noisy or inaccurate connections among nodes, where two nodes with different ground-truth labels keep linkage relations. These unreliable relationships would confuse the network learning and cause biased representations, then lead to sub-optimal performance of clustering. To solve this problem, the random walk mechanism is introduced to adjacent matrix reliability and abundance enhancement by measuring localized structure similarities among nodes. It can filter out noisy connections and add some reliable connections in the original graph, simultaneously, as shown in Fig. 3. Specifically, we first construct t random walk matrices $\mathbf{F} = \{\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_t\}$ over the original graph by conducting k-step random walk starting from central node for t times, where $\mathbf{F}_i \in \mathbb{R}^{N\times N}$, k and t are two pre-defined hyper-parameters. Then we obtain the averaged random walk matrix $\bar{\mathbf{F}} = \{f_1, f_2, \ldots, f_N\} \in \mathbb{R}^{N\times N}$ by Eq. (1) and utilize it to construct the weighted adjacent matrix. By doing this, $\bar{\mathbf{F}}$ could well extract and preserve k-order neighborhood information for each central node.

$$\bar{\mathbf{F}} = \frac{\sum_{i=1}^{t} \mathbf{F_i}}{t}. \tag{1}$$

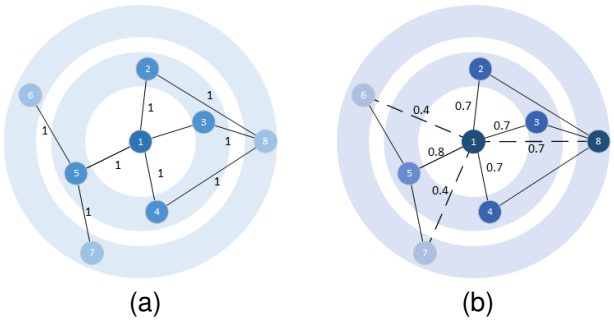

Fig. 3. Illustration of the hidden connection recovery capability of the random walk-based adjacent matrix enhancement strategy. In this figure, by conducting random walk on the original adjacent matrix (a), more abundant connection information would be added as shown in (b). The two large translucent concentric circles indicate the first-order connection and second-order connections among vertexes.

After that, we generate the weighted adjacent matrix $\tilde{\mathbf{A}} = (\tilde{a}_{ij})_{N\times N} \in \mathbb{R}^{N\times N}$ upon $\bar{\mathbf{F}}$ via Eq. (2), where $\tilde{a}_{ij} = f_i f_j^T$ indicates the localized structure (i.e., k-order neighbor relationships) similarity of node i and j. The constructed weighted adjacent matrix $\tilde{\mathbf{A}}$ has two merits as below. On the one hand, it explores more underlying informative information from high-order samples to construct a more informative graph structure. On the other hand, it can filter out the noisy connections to avoid learning the biased representations.

$$\tilde{\mathbf{A}} = \bar{\mathbf{F}}\bar{\mathbf{F}}^T. \tag{2}$$

C. Transformer-based Graph Auto-Encoder (TGAE)

In most current GCN-based auto-encoders, the neighbor information aggregation process can be shown as Eq. (3):

$$\mathbf{Z}^{(l)} = \sigma(\mathbf{D}^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})\mathbf{D}^{-\frac{1}{2}}\mathbf{Z}^{(l-1)}\mathbf{W}^{(l)}), \tag{3}$$

where l indicates the l-th layer, and the non-linear activation function σ can be Tanh or ReLU. The identity matrix $\mathbf{I} \in \mathbb{R}^{N\times N}$ denotes the self-loop for each node,

$\mathbf{D} = diag(d_1, d_2, \ldots, d_N) \in \mathbb{R}^{N\times N}$, and $d_i = \sum_{j=1}^{N} a_{ij}$. $\mathbf{W}^{(l)}$ is the learnable weight matrix of l-th layer, $\mathbf{Z}^{(l)}$ is the latent representation of l-th layer. Eq. (3) implies that the first-order neighbor information plays an significant role in representations learning. Following this principle, the non-local information can not be captured and preserved in the shallow layers, which would lead to learn the less expressive representation and cause unsatisfied clustering performance. Thus, we propose a transformer-based graph auto-encoder (TGAE), which can capture high-order relationships in the latent space in each layer. We will analyze the encoder and decoder of TGAE in the following.

Graph Encoder of TGAE. To solve the aforementioned issue, we introduce the transformer mechanism [26] into the graph auto-encoder to conduct non-local self-attention learning. This learning procedure mainly includes three steps:

1) Sample correlation-based topology generation. Following the transformer architecture [26], we obtain the key matrix $\mathbf{K}^{(l)} \in \mathbb{R}^{N\times d^{(l)}}$ and the query matrix $\mathbf{Q}^{(l)}$ of l-th layer by calculating Eq. (4):

$$\begin{aligned} \mathbf{K}^{(l)} &= \sigma(\mathbf{Z}^{(l-1)}\mathbf{W}_k^{(l)} + \mathbf{b}_k^{(l)}), \\ \mathbf{Q}^{(l)} &= \sigma(\mathbf{Z}^{(l-1)}\mathbf{W}_q^{(l)} + \mathbf{b}_q^{(l)}), \end{aligned} \tag{4}$$

where $\mathbf{W}^{(l)} \in \mathbb{R}^{d^{(l-1)}\times d^{(l)}}$ and $\mathbf{b}^{(l)} \in \mathbb{R}^{1\times d^{(l)}}$ denote the learnable network parameters of l-th layer. σ is the non-linear activation operation, i.e., Tanh. Then we calculate the dot-product attention between each two nodes with Eq. (5) to generate the fully-connected graph $\mathbf{G}^{(l)} \in \mathbb{R}^{N\times N}$:

$$\mathbf{G}^{(l)} = exp\left(\frac{\mathbf{Q}^{(l)}\mathbf{K}^{(l)T}}{\sqrt{d^{(l)}}}\right). \tag{5}$$

2) Two-source structural information fusion. The learned sample correlation-based topology $\mathbf{G}$ depends only on the feature similarity of sample-wise pairs. To further explore more accurate linkage relationships and improve their reliability, we regard the normalized adjacent matrix $\tilde{\mathbf{A}}$ as prior information and introduce it to refine the $\mathbf{G}$. Specifically, we combine $\tilde{\mathbf{A}}$ with $\mathbf{G}$ via a Hadamard product operation, as formulated:

$$\tilde{\mathbf{G}}_{ij}^{(l)} = \frac{\tilde{\mathbf{A}}_{ij}\mathbf{G}_{ij}^{(l)}}{\sum_{n=1}^{N}\tilde{\mathbf{A}}_{in}\mathbf{G}_{in}^{(l)}}, \tag{6}$$

where $\tilde{\mathbf{G}}$ indicates the fused topology matrix associated with both structure and attribute information.

3) Graph encoding via transformer. After that, we take the key matrix $\mathbf{K}^{(l)}$ as the inputs and obtain the value matrix $\mathbf{V}^{(l)} \in \mathbb{R}^{N\times d^{(l)}}$ of l-th layer by calculating Eq. (7):

$$\mathbf{V}^{(l)} = \sigma(\mathbf{K}^{(l)}\mathbf{W}_v^{(l)} + \mathbf{b}_v^{(l)}), \tag{7}$$

where $\mathbf{W}_v^{(l)} \in \mathbb{R}^{d^{(l)}\times d^{(l)}}$ and $\mathbf{b}_v^{(l)} \in \mathbb{R}^{1\times d^{(l)}}$ denote the learnable network parameters of l-th layer. σ is the non-linear activation operation, i.e., Tanh. In addition, we extract the latent embedding matrix $\mathbf{Z}^{(l)} \in \mathbb{R}^{N\times d^{(l)}}$ of l-th layer:

$$\mathbf{Z}^{(l)} = \tilde{\mathbf{A}}(\tilde{\mathbf{G}}^{(l)}\mathbf{V}^{(l)}). \tag{8}$$

In contrast with the GAE in DFCN [25], the proposed TGAE enables to explore the first-order and the long-range relationships in each layer which improves the discrimination of the learned representations.

Graph Decoder of TGAE. Following DFCN [25], we integrate the attribute embedding $\mathbf{Z}_{AE} \in \mathbb{R}^{N\times d'}$ and the graph embedding $\mathbf{Z}_{TGAE} \in \mathbb{R}^{N\times d'}$ learned from the corresponding AE and TGAE to obtain the initially fused embedding $\mathbf{Z}_I \in \mathbb{R}^{N\times d'}$ by calculating Eq. (9), and d' is the dimension of the fused embedding.

$$\mathbf{Z}_I = \alpha\mathbf{Z}_{AE} + (1-\alpha)\mathbf{Z}_{TGAE}, \tag{9}$$

where α denotes a learnable parameter to balance the importance of $\mathbf{Z}_{AE}$ and $\mathbf{Z}_{TGAE}$. Finally, we obtain the enhanced fused embedding by calculating Eq. (10):

$$\tilde{\mathbf{Z}} = \tilde{\mathbf{A}}\mathbf{Z}_I, \tag{10}$$

where $\tilde{\mathbf{Z}}$ is the input fed into the two decoders of AE and TGAE.

Note that our proposed auto-encoder-based framework is symmetrical, thus in the decoder, the representation learning process in the h-th layer is formulated as:

$$\hat{\mathbf{Z}}^{(h)} = \tilde{\mathbf{A}}(\hat{\mathbf{G}}^{(h)}\hat{\mathbf{V}}^{(h)}), \tag{11}$$

where $\hat{\mathbf{G}}^{(h)} \in \mathbb{R}^{N\times N}$ and $\hat{\mathbf{V}}^{(h)} \in \mathbb{R}^{N\times d^{(h)}}$ denote the fused topology matrix and the value matrix of h-th layer in the graph decoder, respectively.

D. Loss Function and Training

In our proposed DFCN-RSP, the overall loss function contains two parts, i.e., the reconstruction and clustering processes:

$$L = L_{AE} + L_{TGAE} + \beta L_{KL}, \tag{12}$$

where L_{AE}, L_{TGAE}, and L_{KL} indicate the information reconstruction of AE and TGAE, and the clustering loss, respectively. The pre-defined hyper-parameter β is utilized to adjust the importance of the above learning processes. The details are as follows:

1) L_{AE} is the reconstruction loss between raw node attributes $\mathbf{X}$ and rebuilt node attributes $\hat{\mathbf{X}} \in \mathbb{R}^{N\times d}$ of AE as Eq. (13):

$$L_{AE} = \frac{1}{N}\left\|\mathbf{X} - \hat{\mathbf{X}}\right\|_F^2, \tag{13}$$

where N is the number of nodes.

2) L_{TGAE} includes reconstruction losses of both node attributes and graph structure:

$$L_{TGAE} = \frac{1}{2N}\left\|\tilde{\mathbf{A}}\mathbf{X} - \hat{\mathbf{Z}}\right\|_F^2 + \frac{\theta}{2N}\left\|\tilde{\mathbf{A}} - \hat{\mathbf{A}}\right\|_F^2, \tag{14}$$

where θ is a pre-defined hyper-parameter that balances the weights of the two parts in Eq. (14), $\hat{\mathbf{Z}} \in \mathbb{R}^{N\times d}$ is the reconstructed weighted attribute matrix of TGAE, and the rebuilt adjacent matrix $\hat{\mathbf{A}} \in \mathbb{R}^{N\times N}$ is obtained via a simple decoding operation (e.g., inner product module).

3) L_{KL} is a triplet clustering loss among three soft assignment matrices $\mathbf{B}, \mathbf{B}', \mathbf{B}'' \in \mathbb{R}^{N\times K}$ for fusion structure, TGAE

and AE, respectively, with target distribution $\mathbf{P} \in \mathbb{R}^{N\times K}$ which is generated from $\mathbf{B}$ by adapting the KL-divergence:

$$b_{ij} = \frac{(1+\|\tilde{z}_i - u_j\|^2/v)^{-\frac{v+1}{2}}}{\sum_{j'}(1+\|\tilde{z}_i - u_{j'}\|^2/v)^{-\frac{v+1}{2}}}, \tag{15}$$

$$p_{ij} = \frac{b_{ij}^2/\sum_i b_{ij}}{\sum_{j'}(b_{ij'}^2/\sum_i b_{ij'})}, \tag{16}$$

$$L_{KL} = \sum_{i=1}^{N}\sum_{j=1}^{K} p_{ij} log \frac{p_{ij}}{(b_{ij}+b'_{ij}+b''_{ij})/3}. \tag{17}$$

In Eq. (15), $\tilde{z}_i$ is the fused embedding of node i, and u_j is the j-th pre-calculated clustering center. By using the student's t-distribution as kernel, the similarity between $\tilde{z}_i$ and u_j can be calculated, and v is the degree of freedom. b_{ij} indicates the soft assignment that assigns node i to the j-th center, and the matrix $\mathbf{B}$ reflects the distribution of all samples. Similarly, $\mathbf{B}'$ and $\mathbf{B}''$ are the corresponding soft assignment matrices for $\mathbf{Z}_{TGAE}$ and $\mathbf{Z}_{AE}$. Then the generated target distribution $\mathbf{P}$ can be derived from Eq. (16), and the final clustering loss can be calculated by Eq. (17). This triplet clustering loss objective aims to conduct the distribution alignment among the robust target distribution $\mathbf{P}$, the soft assignment distribution $\mathbf{B}$, $\mathbf{B}'$, and $\mathbf{B}''$ at the same time. More details can be found in DFCN [25] and we summarize the learning procedure of the proposed DFCN-RSP in Algorithm 1.

Algorithm 1 The training procedure of DFCN-RSP

Input: Attribute matrix $\mathbf{X}$; original adjacent matrix $\mathbf{A}$; clusters K; random walk step k; random walk times t; iteration numbers O; target distribution update interval U; balanced coefficients θ, β.
Output: Clustering results $\mathbf{R}$.

```
Repeat t times k-step random walk based on A to obtain F = {F_1, F_2, ..., F_t} and compute the weighted adjacent matrix Ã via Eq. (1) and Eq. (2);
Pre-train TGAE to obtain Z_TGAE by calculating Eq. (4) - (8);
Pre-train AE to obtain Z_AE;
Obtain the fused representations Z̃ by Eq. (9) and Eq. (10);
Obtain the initialized clustering centers u by conducting K-means over Z̃;
for o = 1 to O do
    Update Z_I and Z̃ by corresponding Eq. (9) and Eq. (10);
    Obtain B, B', B'' based on Z̃, Z_TGAE and Z_AE, respectively by Eq. (15) ;
    if o % U == 0 then
        Obtain the target distribution P derived from B by Eq. (16);
    end if
    Utilize P to refine B, B', B'' by Eq. (17);
    Minimize Eq. (12) to optimize the whole framework.
end for
Perform K-mean over the resultant graph embedding to obtain the final clustering results R.
return R
```

IV. EXPERIMENTS

A. Benchmark Datasets

We evaluate our proposed DFCN-RSP on five well-known public graph datasets including ACM[1], DBLP[2], AMAP [46], PUBMED [47], and COREFULL [48]. The brief dataset descriptions are illustrated in Table II.

TABLE II
DATASETS INTRODUCTION.

Dataset	#Samples	#Dimensions	#Edges	#Clusters
ACM	3025	1870	13128	3
DBLP	4057	334	3528	4
AMAP	7650	745	119081	8
PUBMED	19717	500	44325	3
COREFULL	19793	8710	63421	70

B. Experiment Setup

Training Procedure We implement the experiments on the PyTorch platform with one NVIDIA GeForce RTX 3080. For the training procedure, we firstly refine the original adjacent matrix to obtain the weighted adjacent matrix $\tilde{\mathbf{A}}$ via random walk operation. By minimizing Eq. (13) and Eq. (14), we then pre-train AE and TGAE to reconstruct node attributes and graph structure for 30 iterations, respectively. After that, we integrate the two sub-networks into a united framework with an alignment loss function and train it for 30 iterations on PUBMED and CORAFULL, and 50 iterations on the other datasets. Finally, we fine-tune the whole framework until convergence by minimizing Eq. (12). We obtain the cluster-ID by performing K-means over $\tilde{\mathbf{Z}}$. For all experiments, we conduct 10 runs to avoid the adverse effect of randomness, then we report the mean values with corresponding standard deviations.

Evaluation Metrics Four publicly-used metrics are adopted to evaluate the clustering performance [25]:

- Clustering Accuracy (ACC),
- Macro F1-score (F1),
- Average Rand Index (ARI),
- Normalized Mutual Information (NMI).

Baseline Methods All compared baseline algorithms are summarized in the following:

- **K-means** [50] is one of the most classical clustering methods;
- **AE** [51] is a typical auto-encoder based method that maps the original sample attributes into the low-dimension space to obtain the node representations;
- **DEC** [18] **& IDEC** [19] are two representative auto-encoder based deep clustering methods using student's

[1] https://dl.acm.org/
[2] https://dblp.uni-trier.de

TABLE III
CLUSTERING PERFORMANCE OF 15 METHODS ON FIVE DATASETS (MEAN ± STD). THE **BOLD** VALUES INDICATE THE CORRESPONDING BEST RESULTS.

Method	ACM	DBLP	AMAP	PUBMED	CORAFULL
			ACC (%)		
K-means	67.31 ± 0.71	38.65 ± 0.65	27.22 ± 0.76	59.83 ± 0.01	26.27 ± 1.10
AE	81.83 ± 0.08	51.43 ± 0.35	48.25 ± 0.08	63.07 ± 0.31	33.12 ± 0.19
DEC	84.33 ± 0.76	58.16 ± 0.56	47.22 ± 0.08	60.14 ± 0.09	31.92 ± 0.45
IDEC	85.12 ± 0.52	60.31 ± 0.62	47.62 ± 0.08	60.70 ± 0.34	32.19 ± 0.31
GAE	84.52 ± 1.44	61.21 ± 1.22	71.57 ± 2.48	62.09 ± 0.81	29.60 ± 0.81
VGAE	84.13 ± 0.22	58.59 ± 0.06	74.26 ± 3.63	68.48 ± 0.77	32.66 ± 1.29
DAEGC	86.94 ± 2.83	62.05 ± 0.48	76.44 ± 0.01	68.73 ± 0.03	34.35 ± 1.00
ARGA	86.29 ± 0.36	64.83 ± 0.59	69.28 ± 2.30	65.26 ± 0.12	22.07 ± 0.43
ARVGA	83.89 ± 0.54	54.41 ± 0.42	61.46 ± 2.71	64.25 ± 1.24	29.57 ± 0.59
$SDCN_Q$	86.95 ± 0.08	65.74 ± 1.34	35.53 ± 0.39	64.39 ± 0.30	29.75 ± 0.69
SDCN	90.45 ± 0.18	68.05 ± 1.81	53.44 ± 0.81	64.20 ± 1.30	26.67 ± 0.40
MVGRL	86.73 ± 0.76	42.73 ± 1.02	45.19 ± 1.79	67.01 ± 0.52	31.52 ± 2.95
DFCN	90.90 ± 0.20	76.00 ± 0.80	76.88 ± 0.80	68.89 ± 0.07	37.51 ± 0.81
DCRN	91.93 ± 0.20	79.66 ± 0.25	79.94 ± 0.13	**69.87 ± 0.07**	38.80 ± 0.60
Ours	**92.95 ± 0.12**	**81.05 ± 0.02**	**80.61 ± 0.17**	69.30 ± 0.14	**38.81 ± 0.66**
			NMI (%)		
K-means	32.44 ± 0.46	11.45 ± 0.38	13.23 ± 1.33	31.05 ± 0.02	34.68 ± 0.84
AE	49.30 ± 0.16	25.40 ± 0.16	38.76 ± 0.30	26.32 ± 0.57	41.53 ± 0.25
DEC	54.54 ± 1.51	29.51 ± 0.28	37.35 ± 0.05	22.44 ± 0.14	41.67 ± 0.24
IDEC	56.61 ± 1.16	31.17 ± 0.50	37.83 ± 0.08	23.67 ± 0.29	41.64 ± 0.28
GAE	55.38 ± 1.92	30.80 ± 0.91	62.13 ± 2.79	23.84 ± 3.54	45.82 ± 0.75
VGAE	53.20 ± 0.52	26.92 ± 0.06	66.01 ± 3.40	30.61 ± 1.71	47.38 ± 1.59
DAEGC	56.18 ± 4.15	32.49 ± 0.45	65.57 ± 0.03	28.26 ± 0.03	49.16 ± 0.73
ARGA	56.21 ± 0.82	29.42 ± 0.92	58.36 ± 2.76	24.80 ± 0.17	41.28 ± 0.25
ARVGA	51.88 ± 1.04	25.90 ± 0.33	53.25 ± 1.91	23.88 ± 1.05	48.77 ± 0.44
$SDCN_Q$	58.90 ± 0.17	35.11 ± 1.05	27.90 ± 0.40	26.67 ± 1.31	40.10 ± 0.22
SDCN	68.31 ± 0.25	39.50 ± 1.34	44.85 ± 0.83	22.87 ± 2.04	37.38 ± 0.39
MVGRL	60.87 ± 1.40	15.41 ± 0.63	36.89 ± 1.31	31.59 ± 1.45	48.99 ± 3.95
DFCN	69.40 ± 0.40	43.70 ± 1.00	69.21 ± 1.00	31.43 ± 0.13	51.30 ± 0.41
DCRN	71.56 ± 0.61	48.95 ± 0.44	**73.70 ± 0.24**	32.20 ± 0.08	51.91 ± 0.35
Ours	**74.08 ± 0.33**	**51.25 ± 0.02**	72.44 ± 0.19	**32.95 ± 0.33**	**52.34 ± 0.23**
			ARI (%)		
K-means	30.60 ± 0.69	6.97 ± 0.39	5.50 ± 0.44	28.10 ± 0.01	9.35 ± 0.57
AE	54.64 ± 0.16	12.21 ± 0.43	20.80 ± 0.47	23.86 ± 0.67	18.13 ± 0.27
DEC	60.64 ± 1.87	23.92 ± 0.39	18.59 ± 0.04	19.55 ± 0.13	16.98 ± 0.29
IDEC	62.16 ± 1.50	25.37 ± 0.60	19.24 ± 0.07	20.58 ± 0.39	17.17 ± 0.22
GAE	59.46 ± 3.10	22.02 ± 1.40	48.82 ± 4.57	20.62 ± 1.39	17.84 ± 0.86
VGAE	57.72 ± 0.67	17.92 ± 0.07	56.24 ± 4.66	30.15 ± 1.23	20.01 ± 1.38
DAEGC	59.35 ± 3.89	21.03 ± 0.52	59.39 ± 0.02	29.84 ± 0.04	22.60 ± 0.47
ARGA	63.37 ± 0.86	27.99 ± 0.91	44.18 ± 4.41	24.35 ± 0.17	12.38 ± 0.24
ARVGA	57.77 ± 1.17	19.81 ± 0.42	38.44 ± 4.69	22.82 ± 1.52	18.80 ± 0.57
$SDCN_Q$	65.25 ± 0.19	34.00 ± 1.76	15.27 ± 0.37	24.61 ± 1.46	16.47 ± 0.38
SDCN	73.91 ± 0.40	39.15 ± 2.01	31.21 ± 1.23	22.30 ± 2.07	13.63 ± 0.27
MVGRL	65.07 ± 1.76	8.22 ± 0.50	18.79 ± 0.47	29.42 ± 1.06	19.11 ± 2.63
DFCN	74.90 ± 0.40	47.00 ± 1.50	58.98 ± 0.84	30.64 ± 0.11	24.46 ± 0.48
DCRN	77.56 ± 0.52	53.60 ± 0.46	63.69 ± 0.20	31.41 ± 0.12	25.25 ± 0.49
Ours	**80.15 ± 0.31**	**56.92 ± 0.03**	**64.28 ± 0.32**	**31.66 ± 0.21**	**26.00 ± 0.54**
			F1 (%)		
K-means	67.57 ± 0.74	31.92 ± 0.27	23.96 ± 0.51	58.88 ± 0.01	22.57 ± 1.09
AE	82.01 ± 0.08	52.53 ± 0.36	47.87 ± 0.20	64.01 ± 0.29	28.40 ± 0.30
DEC	84.51 ± 0.74	59.38 ± 0.51	46.71 ± 0.12	61.49 ± 0.10	27.71 ± 0.58
IDEC	85.11 ± 0.48	61.33 ± 0.56	47.20 ± 0.11	62.41 ± 0.32	27.72 ± 0.41
GAE	84.65 ± 1.33	61.41 ± 2.23	68.08 ± 1.76	61.37 ± 0.85	25.95 ± 0.75
VGAE	84.17 ± 0.23	58.69 ± 0.07	70.38 ± 2.98	67.68 ± 0.89	29.06 ± 1.15
DAEGC	87.07 ± 2.79	61.75 ± 0.67	69.97 ± 0.02	68.23 ± 0.02	26.96 ± 1.33
ARGA	86.31 ± 0.35	64.97 ± 0.66	64.30 ± 1.95	65.69 ± 0.13	18.85 ± 0.41
ARVGA	83.87 ± 0.55	55.37 ± 0.40	58.50 ± 1.70	64.51 ± 1.32	25.43 ± 0.62
$SDCN_Q$	86.84 ± 0.09	65.78 ± 1.22	34.25 ± 0.44	65.46 ± 0.39	24.62 ± 0.53
SDCN	90.42 ± 0.19	67.71 ± 1.51	50.66 ± 1.49	65.01 ± 1.21	22.14 ± 0.43
MVGRL	86.85 ± 0.72	40.52 ± 1.51	39.65 ± 2.39	67.07 ± 0.36	26.51 ± 2.87
DFCN	90.80 ± 0.20	75.70 ± 0.80	71.58 ± 0.31	68.10 ± 0.07	31.22 ± 0.87
DCRN	91.94 ± 0.20	79.28 ± 0.26	73.82 ± 0.12	68.94 ± 0.08	31.68 ± 0.76
Ours	**92.97 ± 0.12**	**80.52 ± 0.02**	**74.09 ± 0.13**	**69.11 ± 0.16**	**32.22 ± 0.38**

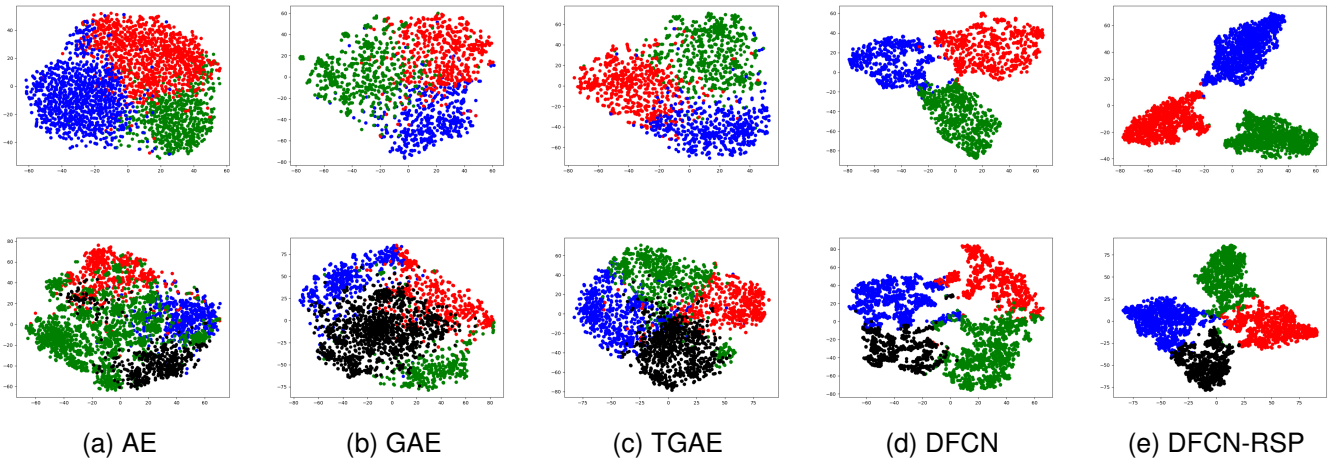

Fig. 4. 2D visualization of results for the compared methods including AE, GAE, TGAE, DFCN and DFCN-RSP by the t-SNE [49] on the ACM and DBLP datasets, respectively.

t-distribution;

- **GAE & VGAE** [27] are two typical graph convolutional network based methods that learn clustering representations via involving structural information;
- **ARGA & ARVGA** [29] learn node embeddings for clustering via the adversarial learning technique;
- **DAEGC** [20] is a task-oriented graph clustering method that integrates the processes of representation learning and clustering into a united framework;
- **MVGRL** [52] is a multi-view graph representation learning method that leverages the cross-view contrastive learning strategy for clustering;
- **SDCN$_Q$ & SDCN** [24] are typical deep graph clustering methods that make full use of structure and attribute information;
- **DFCN** [25] is the original version of our proposed clustering method;
- **DCRN** [31] is a newly published contrastive learning based graph clustering method that makes the first time to consider representation collapse issue in clustering.

Parameter Settings For ARVGA, MVGRL, DFCN, and DCRN, we follow their original settings in the corresponding papers and report the reproduced clustering results, respectively. For the others, we report the clustering results in the paper DFCN [25] directly. For our proposed DFCN-RSP, we set the learning rate to 5e-4 for ACM, DBLP, and AMAP, and 1e-5 for PUBMED and CORAFULL. We fix the two balanced hyper-parameters θ and β to 0.1 and 1. For the learning process of random walk, we conduct a 2-step random walk operation 20 times for all datasets. We train all compared methods with the Adam optimizer.

C. Clustering Performance Comparison

In Table III, we show the clustering performance on five benchmark datasets of fifteen algorithms in terms of four metrics. From these clustering results, we can conclude several observations as below:

- Our proposed DFCN-RSP consistently achieves superior performance against other baseline methods on all datasets. It achieves 2.59%, 3.32%, 0.59%, 0.25%, and 0.75% ARI increments on all datasets compared with the sub-optimal methods. This is because we provide a more informative graph structure and the improved GCNs, leading to representation learning performance enhancement.
- Compared with the methods that exploit the node attributes but overlook the structural information, such as K-means, AE, DEC, and IDEC, our proposed DFCN-RSP sufficiently integrates the two-source information from both graph structure and node attributes, thus achieving better clustering performance.
- Our method also outperforms the GCN-based methods including DAEGC, ARGA/ARVGA, and GAE/VGAE, which is attributed to the effectiveness of our proposed improved graph auto-encoder, i.e., TGAE. In our designed TGAE, the information propagation among long-range nodes benefits the network learning from the perspectives of performance and efficiency.
- Compared to the strongest baselines, such as SDCN/SDCN$_Q$, MVGRL, DCRN, and DFCN, DFCN-RSP achieves better clustering performance on most of the used datasets. This is because all of the above methods ignore the noisy or inaccurate connections in the original graph, which would confuse the network learning and cause biased representations, thus leading to unsatisfied clustering performance. In contrast, DFCN-RSP adopts an improved adjacent matrix, which could filter out the inaccurate connections as well as preserve the reliable connections.

D. Visualization of Clustering Results

In Fig. 4, we illustrate the distribution of learned clustering embeddings in 2D space with t-SNE algorithm [49] on ACM and DBLP datasets. Our proposed DFCN-RSP could better

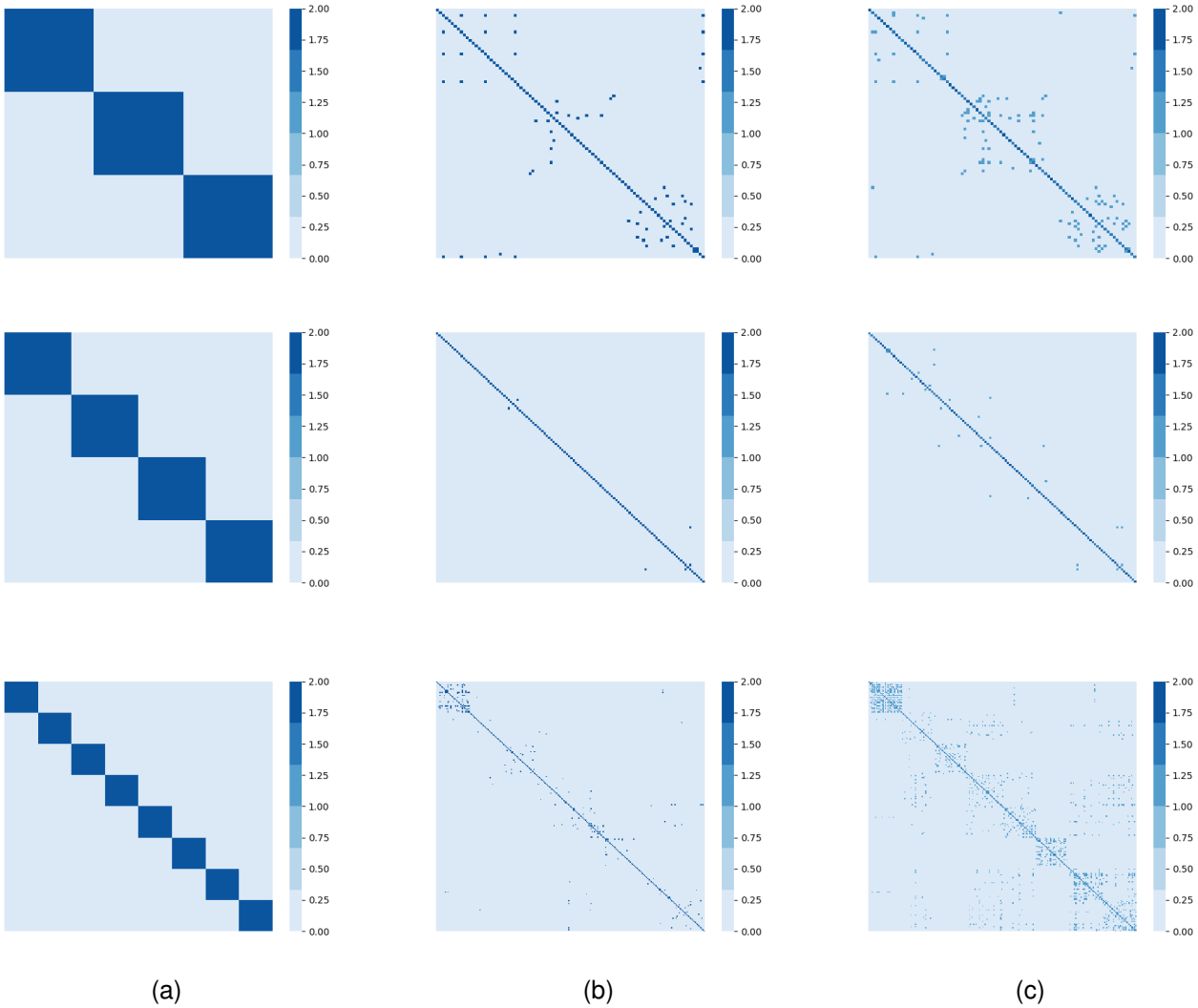

Fig. 5. Illustration of (a) the label matrix induced similarity matrix, (b) raw adjacent matrix, (c) random walk enhanced adjacent matrix on the ACM, DBLP and AMAP dataset, respectively.

reflect the intrinsic clustering structure among data compared with other baselines.

E. Ablation Studies

Effectiveness of the Weighted Adjacent Matrix We further verify the effectiveness of the weighted adjacent matrix $\tilde{\mathbf{A}}$ learned by the random walk mechanism via ablation analysis. It is worth noting that we adopt the 2-step random walk 20 times in experiments for a fair comparison.

1) In Fig. 5, we visualize the label matrices induced similarity matrices, raw adjacent matrices, and our weighted adjacent matrices on ACM, DBLP, and AMAP. Specifically, we firstly sample 30 nodes from each cluster to obtain the corresponding sub-adjacent matrices. Then, to clearly illustrate this point, we add 1 to the positive values in every matrix, thus the numbers of relationship weights vary from 0 to 2. From the matrices in column (b), we can find that the raw graphs contain some edges among nodes in different clusters yet with the same weight, which would reduce the

TABLE IV
STATISTICAL COMPARISON BETWEEN THE WEIGHTED ADJACENT MATRICES $\tilde{\mathbf{A}}$ AND THE RAW ADJACENT MATRICES $\mathbf{A}$ ON THREE DATASETS. TO REVEAL THE EFFECTIVENESS OF THE RANDOM WALK-BASED AFFINITY MATRIX ENHANCEMENT OPERATION, WE FIRST CONNECT ALL VERTEXES THAT BELONG TO THE SAME CLUSTER, CUT ALL OTHER CONNECTIONS, AND RECORD THE CONNECTION NUMBER AS n. n_1 IS THE CORRECT EDGE NUMBER THAT CONNECTS SAMPLES FROM THE SAME CLUSTER ON THE CORRESPONDING ADJACENT MATRIX. WE COUNT AN EDGE AS ONE IF THE EDGE'S WEIGHT IS POSITIVE.

Dataset		n	n_1	$\frac{n_1}{n}$
ACM	A	3054947	24575	0.80%
	$\tilde{A}$	3054947	51053	1.67%
DBLP	A	4229751	9593	0.23%
	$\tilde{A}$	4229751	23979	0.57%
AMAP	A	9642486	204668	2.12%
	$\tilde{A}$	9642486	1049732	10.89%

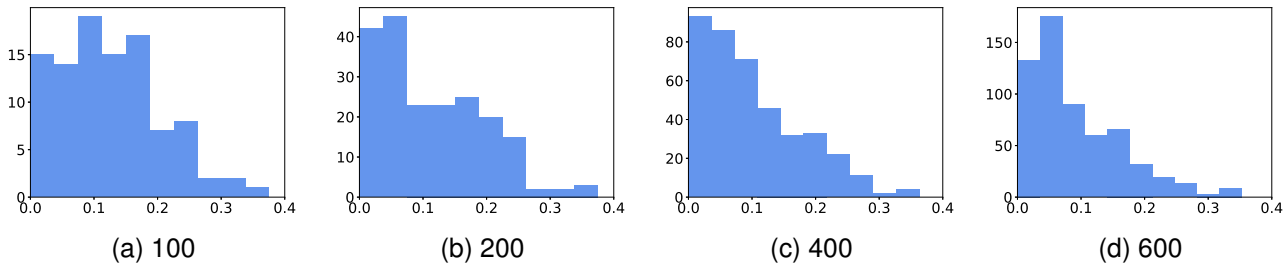

Fig. 6. Evaluation of the robustness of the random walk affinity matrix enhancement operation. In this experiment, we add 100, 200, 400, 600 random connections which connect samples from different clusters as noisy connections. We set all initial connection weights as 1. In the figures, we illustrate the histogram of connection weight after the random walk operation. As we can see, most of the weights of noisy connections are reduced to an acceptable magnitude, which would make the corresponding algorithm more robust to the noisy connections within the data.

efficiency of representation learning. However, our weighted adjacent matrices in column (c) could provide amounts of reliable edges, and filter out the noisy edges or decrease their weights, simultaneously. Although there are also some introduced noises in our matrices, their weights seldom affect the clustering performance, which can be illustrated via the corresponding experiments. Also, in Table IV, we make a statistical comparison between the weighted adjacent matrices $\tilde{\mathbf{A}}$ and the raw adjacent matrices $\mathbf{A}$ on three datasets. To reveal the effectiveness of the random walk-based affinity matrix enhancement operation, we first connect all vertexes that belong to the same cluster while cutting other connections. Here, the connection number is recorded as n. Besides, n_1 is the correct edge number that connects samples from the same cluster on the corresponding adjacent matrix. We count an edge as one if the edge's weight is positive. It directly verifies the effectiveness of our weighted adjacent matrix to address the sparsity and inaccuracy of the raw graph.

2) We add different numbers of noisy edges randomly among nodes in disjoint clusters into the ACM dataset to further verify the robustness of our weighted adjacent matrix refined by the random walk mechanism. We plot the histogram statistics of these added edges and illustrate them in Fig. 6. Specifically, Fig. (a)-(d) are the corresponding histograms for 100, 200, 400, and 600 wrong edges, the abscissa indicates the weight of added edges and the ordinate indicates the corresponding number. We find the weights of the noisy edges are from 0 to 0.35 and most of them are less than 0.1, which can verify the robustness of our improved adjacent matrix.

3) We compare the performance of clustering between baseline GAE in DFCN [25] and GAE-R, which have the same settings but different adjacent matrices. Specifically, the former uses the general normalized adjacent matrix in DFCN and the latter uses the corresponding weighted adjacent matrix refined by the random walk mechanism. The results in Fig. 7 show that the clustering performance of GAE-R is consistently improved on all datasets on most of the metrics, and it also verifies the effectiveness of the weighted adjacent matrix used in our method.

Effectiveness of the Attention Layer In this subsection, we verify the effectiveness of the added attention layers in our method.

1) The results of corresponding ablation studies are also reported in Fig. 7, the GAE-T, which is enhanced by the transformer mechanism, outperforms GAE on most metrics on the five datasets.

TABLE V
ABLATION STUDY OF THE RANDOM WALK AND THE TRANSFORMER MECHANISMS. BASELINE INDICATES THE DFCN [25] ALGORITHM, BASELINE-R INDICATES THE BASELINE WITH THE RANDOM WALK MECHANISM, BASELINE-T INDICATES THE TRANSFORMER ENHANCED BASELINE, AND DFCN-RSP IS THE PROPOSED ALGORITHM.

Dataset	Metric	Baseline-R	Baseline-T	DFCN-RSP
ACM	ACC	92.94 ± 0.11	91.26 ± 0.12	**92.95 ± 0.12**
	NMI	74.06 ± 0.34	70.30 ± 0.34	**74.08 ± 0.33**
	ARI	80.12 ± 0.29	75.95 ± 0.31	**80.15 ± 0.31**
	F1	92.95 ± 0.11	91.24 ± 0.12	**92.97 ± 0.12**
DBLP	ACC	79.47 ± 0.07	80.45 ± 0.03	**81.05 ± 0.02**
	NMI	48.95 ± 0.15	50.11 ± 0.05	**51.25 ± 0.02**
	ARI	54.33 ± 0.15	55.69 ± 0.05	**56.92 ± 0.03**
	F1	78.81 ± 0.07	79.81 ± 0.03	**80.52 ± 0.02**
AMAP	ACC	79.97 ± 0.15	80.06 ± 0.14	**80.61 ± 0.03**
	NMI	**72.56 ± 0.13**	72.46 ± 0.14	72.44 ± 0.04
	ARI	63.14 ± 0.28	63.72 ± 0.24	**64.28 ± 0.12**
	F1	73.74 ± 0.10	73.49 ± 0.12	**74.09 ± 0.13**
PUBMED	ACC	67.81 ± 0.21	67.49 ± 0.28	**69.30 ± 0.14**
	NMI	31.17 ± 0.47	31.20 ± 0.62	**32.95 ± 0.33**
	ARI	29.85 ± 0.45	29.66 ± 0.56	**31.66 ± 0.21**
	F1	67.49 ± 0.19	67.86 ± 0.29	**69.11 ± 0.16**
CORAFULL	ACC	37.14 ± 1.00	34.62 ± 0.62	**38.79 ± 0.66**
	NMI	52.20 ± 0.19	49.77 ± 0.41	**52.34 ± 0.23**
	ARI	25.18 ± 0.83	21.83 ± 0.44	**26.00 ± 0.54**
	F1	30.95 ± 0.71	28.15 ± 0.65	**32.22 ± 0.38**

2) We also verify the effectiveness of the multi-head mechanism of the transformer in our method and the results are shown in Fig. 8, where L111, L411, L821 denote the numbers of attention-head in the corresponding three layers in TGAE, respectively. Specifically, L821 means that there are 8 attention-heads in the first layer, 2 in the second layer, and 1 in the third layer. The results reflect that TGAE can capture more useful information with the multi-head attention mechanism to improve the clustering performance. In our method, we adopt L411 for ACM, DBLP, and AMAP, and L111 for PUBMED and CORAFULL.

Effectiveness of the DFCN-RSP Here we further analyze the effectiveness of the proposed DFCN-RSP. Firstly, in Fig. 7, GAE-R-T, i.e., TGAE, develops improved performance on

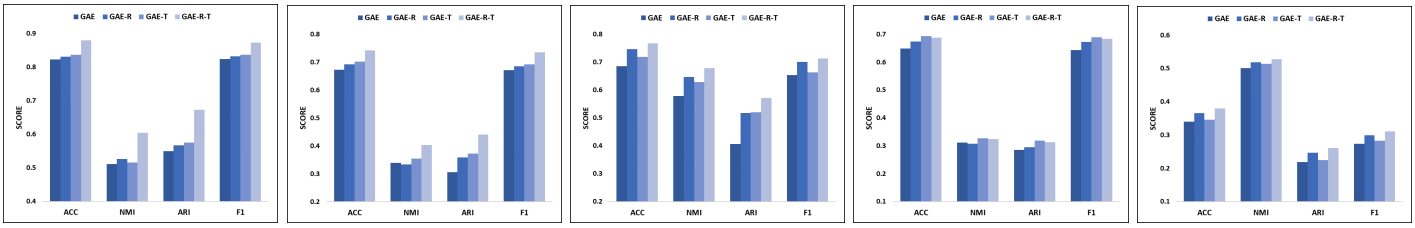

Fig. 7. Effectiveness of random walk and transformer on improving the performance of GAE. In the figures, GAE indicates the baseline graph auto-encoder algorithm in DFCN [25], GAE-R indicates GAE with the random walk mechanism, GAE-T indicates transformer enhanced GAE. GAE-R-T indicates the GAE enhanced by both random walk and transformer.

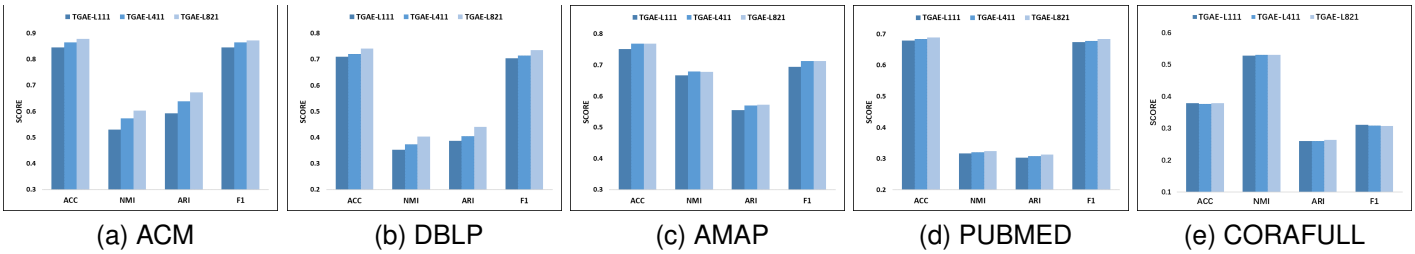

Fig. 8. Illustration of the effect of attention-head numbers. In these figures, take network L821 as an example, it indicates that the transformer in the first, second and third layers possesses 8, 2, 1 head, respectively. ACC, NMI, ARI, and F1 are reported.

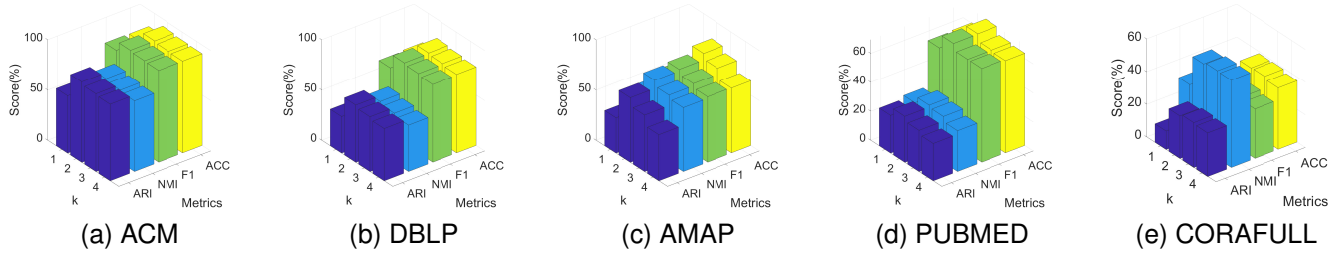

Fig. 9. The sensitivity analysis of DFCN-RSP with the variation of random walk step k.

all datasets with four metrics than the original GAE, which reflects that the proposed TGAE achieves clustering performance enhancement. Furthermore, we compare DFCN-RSP with Baseline-R and Baseline-T in Table V, where the baseline is the DFCN algorithm [25], Baseline-R indicates the baseline with the random walk mechanism, and Baseline-T indicates the transformer enhanced baseline. From these results, we can observe that the random walk mechanism and the transformer mechanism both achieve clustering performance enhancement. We conclude two reasons: 1) the weighted adjacent matrix constructed by the random walk mechanism can filter out the noisy edges and preserve the reliable edges, which provides a more informative graph structure to learn more discriminative representations for clustering; 2) the improved TGAE can capture not only the first-order neighbor but also the high-order relationships in each layer to preserve more informative information.

F. Analysis of Hyper-parameter k

In our method, the weighted adjacent matrix is constructed by the random walk mechanism, and the number of walk steps k is a hyper-parameter to be tuned. To analyze the robustness of DFCN-RSP to k, we investigate the sensitivity analysis of this hyper-parameter. In Fig. 9, we illustrate the clustering performance variation. We can observe that 1) the hyper-parameter k plays an important role in the effectiveness of the weighted adjacent matrix, and then influences the performance of clustering; 2) the clustering performance is poor when $k = 1$, and it first tends to increase and then drops gradually; 3) DFCN-RSP performs better on all datasets when k is set to 2.

V. Conclusion

In this work, we develop a novel deep graph clustering algorithm termed deep fusion clustering network with reliable structure preservation (DFCN-RSP). In this method, we propose to refine the original adjacent matrix by computing the localized structural similarities among nodes with the random walk mechanism. In this way, the noisy or inaccurate edges in the original graph can be filtered and the reliable edges can be supplied and preserved, respectively. Furthermore, we propose a transformer-based graph auto-encoder, which can explore the first-order neighbor information as well as the high-order dependencies in each layer to learn high-quality representations

for improving the clustering performance. Through extensive experiments on several benchmarks on clustering tasks, we empirically verify that our proposed algorithm is superior to the existing ones. In the future, we will further extend our proposed method to handle the multi-view deep clustering task.

ACKNOWLEDGMENTS

This work was supported by the National Key R&D Program of China (project no. 2020AAA0107100) and the National Natural Science Foundation of China (project no. 61922088, 61906020, and 61773392).

REFERENCES

[1] P. Hu, K. C. Chan, and T. He, "Deep graph clustering in social network," in *Proceedings of the 26th International Conference on World Wide Web Companion*, 2017, pp. 1425–1426.

[2] A. Markovitz, G. Sharir, I. Friedman, L. Zelnik-Manor, and S. Avidan, "Graph embedded pose clustering for anomaly detection supplementary material."

[3] Z. Wang, L. Zheng, Y. Li, and S. Wang, "Linkage based face clustering via graph convolution network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1117–1125.

[4] X. Yang, C. Deng, F. Zheng, J. Yan, and W. Liu, "Deep spectral clustering using dual autoencoder network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4066–4075.

[5] U. Shaham, K. Stanton, H. Li, B. Nadler, R. Basri, and Y. Kluger, "Spectralnet: Spectral clustering using deep neural networks," *arXiv preprint arXiv:1801.01587*, 2018.

[6] Z. Huang, J. T. Zhou, H. Zhu, C. Zhang, J. Lv, and X. Peng, "Deep spectral representation learning from multi-view data," *IEEE Transactions on Image Processing*, vol. 30, pp. 5352–5362, 2021.

[7] X. Peng, J. Feng, S. Xiao, W.-Y. Yau, J. T. Zhou, and S. Yang, "Structured autoencoders for subspace clustering," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 5076–5086, 2018.

[8] J. Lv, Z. Kang, X. Lu, and Z. Xu, "Pseudo-supervised deep subspace clustering," *IEEE Transactions on Image Processing*, 2021.

[9] Y. Qin, H. Wu, X. Zhang, and G. Feng, "Semi-supervised structured subspace learning for multi-view clustering," *IEEE Transactions on Image Processing*, 2021.

[10] L. Zhou, B. Xiao, X. Liu, J. Zhou, E. R. Hancock *et al.*, "Latent distribution preserving deep subspace clustering," in *28th International Joint Conference on Artificial Intelligence*. York, 2019, pp. 4440–4446.

[11] P. Ji, T. Zhang, H. Li, M. Salzmann, and I. Reid, "Deep subspace clustering networks," *arXiv preprint arXiv:1709.02508*, 2017.

[12] X. Peng, J. Feng, J. Lu, W.-Y. Yau, and Z. Yi, "Cascade subspace clustering," in *Thirty-First AAAI conference on artificial intelligence*, 2017.

[13] X. Ye, J. Zhao, Y. Chen, and L.-J. Guo, "Bayesian adversarial spectral clustering with unknown cluster number," *IEEE Transactions on Image Processing*, vol. 29, pp. 8506–8518, 2020.

[14] S. Mukherjee, H. Asnani, E. Lin, and S. Kannan, "Clustergan: Latent space clustering in generative adversarial networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 4610–4617.

[15] K. Ghasedi, X. Wang, C. Deng, and H. Huang, "Balanced self-paced learning for generative adversarial clustering network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4391–4400.

[16] L. Yang, N.-M. Cheung, J. Li, and J. Fang, "Deep clustering by gaussian mixture variational autoencoders with graph embedding," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[17] J. Chen, L. Milot, H. M. Cheung, and A. L. Martel, "Unsupervised clustering of quantitative imaging phenotypes using autoencoder and gaussian mixture model," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2019, pp. 575–582.

[18] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*. PMLR, 2016, pp. 478–487.

[19] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved deep embedded clustering with local structure preservation." in *Ijcai*, 2017, pp. 1753–1759.

[20] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang, and C. Zhang, "Attributed graph clustering: A deep attentional embedding approach," *arXiv preprint arXiv:1906.06532*, 2019.

[21] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[22] S. Pan, R. Hu, S. F. Fung, G. Long, J. Jiang, and C. Zhang, "Learning graph embedding with adversarial training methods," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2475–2487, 2020.

[23] K. Hassani and A. H. Khasahmadi, "Contrastive multi-view representation learning on graphs," 2020.

[24] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, and P. Cui, "Structural deep clustering network," in *Proceedings of The Web Conference 2020*, 2020, pp. 1400–1410.

[25] W. Tu, S. Zhou, X. Liu, X. Guo, Z. Cai, E. Zhu, and J. Cheng, "Deep fusion clustering network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, 2021, pp. 9978–9987.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv*, 2017.

[27] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.

[28] Z. Tao, H. Liu, J. Li, Z. Wang, and Y. Fu, "Adversarial graph embedding for ensemble clustering," in *International Joint Conferences on Artificial Intelligence Organization*, 2019.

[29] S. Pan, R. Hu, S.-f. Fung, G. Long, J. Jiang, and C. Zhang, "Learning graph embedding with adversarial training methods," *IEEE transactions on cybernetics*, vol. 50, no. 6, pp. 2475–2487, 2019.

[30] F. M. Bianchi, D. Grattarola, and C. Alippi, "Spectral clustering with graph neural networks for graph pooling," in *International Conference on Machine Learning*. PMLR, 2020, pp. 874–883.

[31] Y. Liu, W. Tu, S. Zhou, X. Liu, L. Song, X. Yang, and E. Zhu, "Deep graph clustering via dual correlation reduction," *arXiv preprint arXiv:2112.14772*, 2021.

[32] Y. Li and G. Baciu, "Hsgan: Hierarchical graph learning for point cloud generation," *IEEE Transactions on Image Processing*, vol. 30, pp. 4540–4554, 2021.

[33] 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, 2019.

[34] S. Chen, C. Duan, Y. Yang, D. Li, C. Feng, and D. Tian, "Deep unsupervised learning of 3d point clouds via graph topology inference and filtering," *IEEE Transactions on Image Processing*, vol. 29, pp. 3183–3198, 2020.

[35] J. Song, L. Gao, F. Nie, H. T. Shen, Y. Yan, and N. Sebe, "Optimized graph learning using partial tags and multiple features for image and video annotation," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 4999–5011, 2016.

[36] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs," *IEEE transactions on signal processing*, vol. 61, no. 7, pp. 1644–1656, 2013.

[37] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[38] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under laplacian and structural constraints," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 825–841, 2017.

[39] M. Chen, I. W. Tsang, M. Tan, and T. J. Cham, "A unified feature selection framework for graph embedding on high dimensional data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 6, pp. 1465–1477, 2014.

[40] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, and A. J. Smola, "Distributed large-scale natural graph factorization," in *Proceedings of the 22nd international conference on World Wide Web*, 2013, pp. 37–48.

[41] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Chang, "Network representation learning with rich text information," in *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[42] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

[43] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD*

international conference on Knowledge discovery and data mining, 2014, pp. 701–710.
[44] L. Franceschi, M. Niepert, M. Pontil, and X. He, "Learning discrete structures for graph neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 1972–1982.
[45] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 66–74.
[46] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann, "Pitfalls of graph neural network evaluation," 2018.
[47] G. Namata, B. London, L. Getoor, and Namatag, B. H. and BLONDON and GETOOR and BERT@CS.UMD.EDU, "Query-driven active surveying for collective classification."
[48] Y. Liu, S. Zhou, X. Liu, W. Tu, and X. Yang, "Improved dual correlation reduction network," *arXiv preprint arXiv:2202.12533*, 2022.
[49] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.
[50] J. A. H. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society*, vol. 28, no. 1, pp. 100–108, 1979.
[51] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.
[52] K. Hassani and A. H. Khasahmadi, "Contrastive multi-view representation learning on graphs," in *International Conference on Machine Learning*. PMLR, 2020, pp. 4116–4126.

Lei Gong graduated from Ocean University of China, Qingdao, China. She is now a student in College of Computer, National University of Defense Technology (NUDT), Hunan, China. She is working hard for pursuing her master degree. Her current research interests include unsupervised graph learning, deep graph clustering, and representation learning.

Wenxuan Tu is pursuing his Ph.D. degree in College of Computer, National University of Defense Technology (NUDT), China. His research interests include unsupervised graph learning, deep graph clustering, and image semantic segmentation. He has published several papers in highly regarded journals and conferences such as AAAI, ICML, MM, IEEE T-IP, Information Sciences, etc.

Yue Liu graduated from Northeastern University at Qinhuangdao, Hebei, China. He was recommended for admission to the National University of Defense Technology (NUDT) with excellent grades and technological innovation capability. He is working hard and pursuing his master degree in College of Computer, NUDT, China. His current research interests include graph neural networks, deep clustering and self-supervised learning.

Sihang Zhou received his PhD degree from School of Computer, National University of Defense Technology (NUDT), China. He is now lecturer at College of Intelligence Science and Technology, NUDT. His current research interests include machine learning and medical image analysis. Dr. Zhou has published 40+ peer-reviewed papers, including IEEE T-IP, IEEE T-NNLS, IEEE T-MI, Information Fusion, Medical Image Analysis, AAAI, MICCAI, etc.

Tiantian Xu received a B.E. and M.E. degree in computer applications from Qilu University of Technology in 2012 and 2015, respectively. She received a Ph.D. degree in software engineering from Ocean University of China in 2018. She is a lecturer in Department of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences). Her research interests include pattern recognition, association rules and sequential pattern mining, machine learning and feature selection. She has published research papers at international journals.

Xinwang Liu (SM' 20) received his PhD degree from National University of Defense Technology (NUDT), China. He is now Professor of School of Computer, NUDT. His current research interests include kernel learning and unsupervised feature learning. Dr. Liu has published 70+ peer-reviewed papers, including those in highly regarded journals and conferences such as IEEE T-PAMI, IEEE T-KDE, IEEE T-IP, IEEE T-NNLS, IEEE T-MM, IEEE T-IFS, ICML, NeurIPS, ICCV, CVPR, AAAI, IJCAI, etc. He serves as the associated editor of IEEE TNNLS and Information Fusion Journal. More information can be found at https://xinwangliu.github.io/.