EGRC-Net: Embedding-Induced Graph Refinement Clustering Network

Zhihao Peng, Hui Liu, Yuheng Jia, *Member, IEEE*, and Junhui Hou, *Senior Member, IEEE*

***Abstract*— Existing graph clustering networks heavily rely on a predefined yet fixed graph, which can lead to failures when the initial graph fails to accurately capture the data topology structure of the embedding space. In order to address this issue, we propose a novel clustering network called Embedding-Induced Graph Refinement Clustering Network (EGRC-Net), which effectively utilizes the learned embedding to adaptively refine the initial graph and enhance the clustering performance. To begin, we leverage both semantic and topological information by employing a vanilla auto-encoder and a graph convolution network, respectively, to learn a latent feature representation. Subsequently, we utilize the local geometric structure within the feature embedding space to construct an adjacency matrix for the graph. This adjacency matrix is dynamically fused with the initial one using our proposed fusion architecture. To train the network in an unsupervised manner, we minimize the Jeffreys divergence between multiple derived distributions. Additionally, we introduce an improved approximate personalized propagation of neural predictions to replace the standard graph convolution network, enabling EGRC-Net to scale effectively. Through extensive experiments conducted on nine widely-used benchmark datasets, we demonstrate that our proposed methods consistently outperform several state-of-the-art approaches. Notably, EGRC-Net achieves an improvement of more than 11.99% in Adjusted Rand Index (ARI) over the best baseline on the DBLP dataset. Furthermore, our scalable approach exhibits a 10.73% gain in ARI while reducing memory usage by 33.73% and decreasing running time by 19.71%. The code for EGRC-Net will be made publicly available at https://github.com/ZhihaoPENG-CityU/EGRC-Net.**

***Index Terms*— Geometric structure information, graph refinement, improved approximate personalized propagation of neural predictions, Jeffreys divergence.**

I. Introduction

CLUSTERING is a typical yet challenging machine learning topic with a series of real-world applications, including object detection [1], [2], [3], [4], [5], social network analysis [6], [7], [8], [9], and face recognition [10], [11], [12], [13]. Clustering aims to partition data into different groups based on their intrinsic patterns, such that similar samples are clustered together while dissimilar samples are separated from each other. Due to the powerful embedding learning capability of the convolutional neural network, deep embedding clustering has been widely studied in recent decades. For example, Hinton and Salakhutdinov [14] proposed a deep auto-encoder network (DAE) to focus on the node attribute information for clustering assignments. Xie et al. [15] jointly learned feature representations and clustering assignments to present the deep embedded clustering network (DEC). Guo et al. [16] improved DEC via a reconstruction loss.

Although these DAE-based networks have shown impressive embedding learning capability, they neglect the topology structure information of the input, limiting the clustering performance. To this end, the graph convolution network (GCN) is utilized to conduct the deep embedding clustering by propagating spatial relations with the representations of its neighbors, namely the graph clustering network. For example, Kipf and Welling [17] proposed the graph auto-encoder to learn the graph structure feature. Pan et al. [22] injected an adversarial regularizer into the GCN framework. Zhang et al. [23] used a relaxed K-means [24] to improve the representation learning capability of the GCN-based clustering method. Recently, numerous works have combined DAE and GCN to simultaneously consider node attribute and topology structure information, achieving significant improvement in clustering performance. For instance, Bo et al. [19] integrated the DAE and GCN features via the structural deep clustering network. Peng et al. [20] employed two feature fusion modules to merge the DAE and GCN features. He et al. [21] conducted the feature fusion and graph reconstruction layer by layer. *Nevertheless, all these GCN-based graph clustering networks adopt a predefined graph as a fixed input and may fail to correctly distinguish the samples if the initial graph cannot truly and precisely reflect their topology structures on the embedding space, which is referred to as the fixed graph issue.* Thus, it is expected that the clustering performance can be boosted by performing graph refinement based on the guidance from the learned embedding representation.

In this paper, we propose a novel embedding-induced graph refinement clustering network (**EGRC-Net**) framework to adaptively use the learned embedding for evolving the initial graph, aiming to boost the embedding learning capability of the graph clustering network. Specifically, we first use

Manuscript received 20 February 2023; revised 5 September 2023; accepted 13 November 2023. Date of publication 22 November 2023; date of current version 30 November 2023. This work was supported by the Hong Kong University Grants Committee under Grant UGC/FDS11/E02/22. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ananda S. Chowdhury. *(Corresponding author: Hui Liu.)*

Zhihao Peng and Junhui Hou are with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong (e-mail: zhihapeng3-c@my.cityu.edu.hk; jh.hou@cityu.edu.hk).

Hui Liu is with the School of Computing and Information Sciences, Caritas Institute of Higher Education, Hong Kong (e-mail: hliu99-c@my.cityu.edu.hk).

Yuheng Jia is with the School of Computer Science and Engineering, Southeast University, Nanjing 210096, China, and also with the Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, Nanjing 210096, China (e-mail: yhjia@seu.edu.cn).

Digital Object Identifier 10.1109/TIP.2023.3333557

the vanilla DAE and GCN modules to conduct embedding learning. Subsequently, we explore the local geometric structure information on the embedding space to construct an adjacency matrix. Then, we utilize multilayer perceptron layers and a series of normalization terms to dynamically fuse the constructed graph with the initial one, seeking to preserve the intrinsic semantic structure information. Finally, we minimize the Jeffreys divergence of multiple derived distributions in an unsupervised manner to jointly conduct embedding learning and graph refinement.

Furthermore, we advance EGRC-Net in a scalable manner to handle the practical large-graph problem [23], [25], [26]. Specifically, the weak scalability of EGRC-Net ascribes to the utilization of GCN, since the explicit message-passing of GCN leads to an expensive neighborhood expansion [27]. Thus, we utilize the approximate personalized propagation of neural predictions (APPNP) [28] instead of multi-layer graph convolution to complete the transmission and aggregation of node information on the graph. However, it is still required to predefine a heuristic threshold that determines the teleport probability. Besides, the predefined value also limits its availability since, in real-world clustering tasks, there has little guideline to set a suitable threshold. Thus, we empirically initialize the threshold using a random generator to sample from a uniform distribution on the interval $[0, 1)$ and learn that variable during the network training. To this end, we can achieve information propagation by using the personalized PageRank with a learned threshold value without relying on a predefined heuristic value, namely improved APPNP (IAPPNP). In this way, we obtain the **scalable EGRC-Net**.

In summary, the contributions of our work are as follows:

- We handle the fixed graph issue by dynamically fusing an embedding-induced graph with the initial one, aiming to preserve the intrinsic semantic structure information.
- We design a unified learning framework by minimizing the Jeffreys divergence of multiple derived distributions to jointly conduct embedding learning and graph refinement, mutually benefiting both components.
- We investigate the scalability of the proposed method by utilizing an improved approximate personalized propagation of neural predictions instead of multi-layer graph convolution to complete the transmission and aggregation of node information on the graph.
- Extensive experiments on nine commonly used benchmark datasets demonstrate that the proposed methods consistently outperform several state-of-the-art approaches. In particular, as shown in Figure 1, EGRC-Net improves the ARI by more than 11.99% over the best baseline on DBLP, and the scalable one has a 10.73% gain in ARI while saving 33.73% in memory and 19.71% in running time.

The rest of this paper is organized as follows. We briefly review the related works in Section II and introduce the proposed network in Section III. Afterward, we give the experimental results and analyses in Section IV and conclude this paper in Section V.

Throughout this paper, matrices are denoted by bold upper case letters, vectors by bold lower case letters, and scalars by italic lower case letters, respectively. Given a matrix $\mathbf{B} \in \mathbb{R}^{m\times n}$, $\|\mathbf{B}\|_F$ denotes the Frobenius norm of $\mathbf{B}$, i.e., $\|\mathbf{B}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}\left|b_{i,j}\right|^2}$. Let $\mathbf{X} \in \mathbb{R}^{n\times d}$ be the raw data, $\mathbf{A} \in \mathbb{R}^{n\times n}$ the adjacency matrix of the graph, we summarize the main notations in Table I.

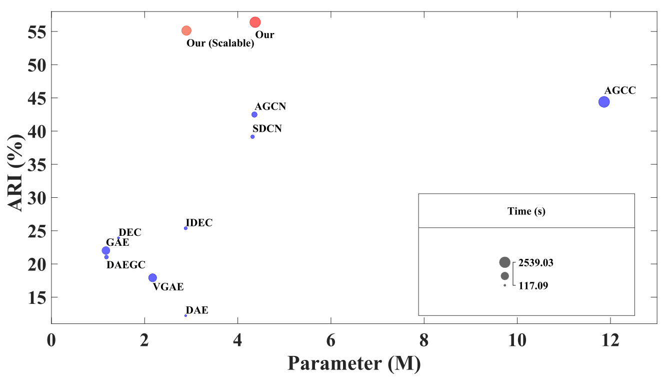


Fig. 1. Comparisons of the adjusted rand index (ARI), parameter numbers (in millions), and running time (in seconds) of different methods on DBLP, where we trained the proposed methods, three DAE-based clustering methods (DAE [14], DEC [15], and IDEC [16]), three GCN-based clustering methods (GAE [17], VGAE [17], and DAEGC [18]), and three DAE and GCN combination-based clustering methods (SDCN [19], AGCN [20], and AGCC [21]) with 200 epochs and repeated experiments ten times. The diameter of the bubble is proportional to the running time, and we make it easier to see all the bubbles by changing the range of diameters to be between 4 and 20 points.

TABLE I
MAIN NOTATIONS AND DESCRIPTIONS

Notations		Descriptions
$\mathbf{X}$	$\in \mathbb{R}^{n\times d}$	The raw matrix
$\hat{\mathbf{X}}$	$\in \mathbb{R}^{n\times d}$	The reconstructed matrix
$\mathbf{A}$	$\in \mathbb{R}^{n\times n}$	The adjacency matrix of the predefined graph
$\mathbf{D}$	$\in \mathbb{R}^{n\times n}$	The degree matrix
$\mathbf{Z}_a$	$\in \mathbb{R}^{n\times \kappa}$	The learned embedding
$\mathbf{A}_z$	$\in \mathbb{R}^{n\times n}$	The constructed adjacency matrix from $\mathbf{Z}_a$
$\mathbf{A}_f$	$\in \mathbb{R}^{n\times n}$	The fused adjacency matrix
$\mathbf{H}$	$\in \mathbb{R}^{n\times d_l}$	The feature extracted from DAE
$\mathbf{Q}$	$\in \mathbb{R}^{n\times \kappa}$	The distribution obtained from $\mathbf{H}$
$\mathbf{P}$	$\in \mathbb{R}^{n\times \kappa}$	The distribution obtained from $\mathbf{Z}_a$
n		The number of samples
κ		The number of clusters
l		The number of encoder/decoder layers
d		The dimension of $\mathbf{X}$
d_l		The dimension of $\mathbf{H}$
ρ		The predefined threshold
Θ		The learned threshold
τ		The number of power iteration steps
$\cdot\|\cdot$		The concatenation operation
$\|\cdot\|_F$		The Frobenius norm

II. Related Work

A. DAE-Based Clustering Methods

Due to the powerful expression ability of the convolutional neural network (CNN), many deep embedding clustering methods have been proposed and achieved impressive performance in practical applications [10], [15], [16], [29], [30], [31]. For example, Hinton and Salakhutdinov [14] adopted a series of CNNs and a reconstruction loss to build a vanilla

DAE, which can be formulated as

$$\min_{\mathbf{H}_i} \left\| \mathbf{X} - \hat{\mathbf{X}} \right\|_F^2 \quad \text{s.t.} \quad \mathbf{H}_0 = \mathbf{X}, \quad \hat{\mathbf{H}}_l = \hat{\mathbf{X}}, \tag{1}$$

where $\hat{\mathbf{X}} \in \mathbb{R}^{n \times d}$, $\mathbf{H}_i = \phi(\mathbf{W}_i^e \mathbf{H}_{i-1} + \mathbf{b}_i^e) \in \mathbb{R}^{n \times d_i}$, and $\hat{\mathbf{H}}_i = \phi(\mathbf{W}_i^d \hat{\mathbf{H}}_{i-1} + \mathbf{b}_i^d) \in \mathbb{R}^{n \times \hat{d}_i}$ indicate the reconstructed data, the i-th encoder and decoder outputs, respectively. $\mathbf{W}_i^e$ and $\mathbf{b}_i^e$, $\mathbf{W}_i^d$ and $\mathbf{b}_i^d$, $\phi(\cdot)$, and $\|\cdot\|_F$ indicate the network weight and bias of the i-th encoder layer, those of the i-th decoder layer, the ReLU activation function, and the Frobenius norm, respectively. Xie et al. [15] designed DEC to improve the DAE framework by learning the embedding and clustering assignments in a joint optimization fashion. Guo et al. [16] introduced a reconstruction loss into the DAE framework to encourage local structure preservation, namely improved deep embedded clustering (IDEC). However, DAE is designed to deal with the grid-wise features of Euclidean space, failing to handle the complex topological data, i.e., non-Euclidean data, in real-world cases.

B. GCN-Based Clustering Methods

GCN integrates the vertex and its neighbors to determine a graph convolution in a spatial-based feature fusion manner, i.e.,

$$\mathbf{Z}_i = \text{LReLU}(\mathbf{D}^{-\frac{1}{2}}(\mathbf{A} + \mathbf{I})\mathbf{D}^{-\frac{1}{2}}\mathbf{Z}'_{\text{i}-1}\mathbf{W}_{\text{i}}), \tag{2}$$

where $\mathbf{Z}'_{i-1} \in \mathbb{R}^{n \times d_{i-1}}$, $\mathbf{Z}_i \in \mathbb{R}^{n \times d_i}$, LReLU, $\mathbf{D} \in \mathbb{R}^{n \times n}$, and $\mathbf{W}_i$ indicate the i-th GCN layer input, the i-th GCN layer output, the Leaky ReLU activation function [32], the degree matrix, and the learnable weight matrix, respectively. $\mathbf{Z}'_0 = \mathbf{X}$. In the recent decade, GCN has rapidly developed in deep graph clustering, crediting from its efficiency, simplicity, and generality [17], [33], [34], [35], [36], [37], [38], [39]. For instance, Kipf and Welling [17] built the graph auto-encoder (GAE) and variational graph auto-encoder (VGAE) based on the variational architecture to learn the graph structure feature via a reconstruction constraint for graph structure preservation. Pan et al. [22] presented the adversarially regularized graph auto-encoder (ARGA) to enhance the learning ability of the network by injecting an adversarial regularizer into the GAE framework. Wang et al. [18] improved GAE by introducing the graph attention network model [40] into its network architecture to encode the topological structure and node contents, called deep attentional embedded graph clustering (DAEGC). Salha et al. [41] designed a general framework (FastGAE) to scale graph AE and VAE to large graphs. Mrabah et al. [42] presented a series of rethinking graph auto-encoder models (e.g., R-GMM-VGAE (RG-VGAE)) to consider the feature randomness and feature drift. Salha-Galvan et al. [43] designed modularity-aware graph autoencoders (MA-GAE) to consider both the initial graph structure and modularity-based prior communities. Zhang et al. [23] developed an embedding graph auto-encoder (EGAE) to use a relaxed K-means [24] to alternatively update GAE by gradient descent and perform clustering on inner-product distance space.

C. DAE and GCN Combination-Based Methods

Recently, numerous works have combined DAE and GCN to achieve clustering [19], [20], [21], [44], [45]. For example, Bo et al. [19] integrated the node attribute and topology structure information based on the DEC framework to design the structural deep clustering network (SDCN). Peng et al. [20] utilized two attention-based feature fusion modules to merge the DAE and GCN features, namely attention-driven graph clustering network (AGCN). He et al. [21] conducted the DAE and GCN feature fusion and the graph construction in each layer to improve the representation ability. Although these approaches have obtained remarkable improvements, they heavily rely on a predefined graph and may fail if the fixed graph cannot well reflect the intrinsic semantic structures on the embedding space. To this end, we propose a novel graph clustering network framework EGRC-Net that adaptively uses the learned embedding to evolve the initial graph.

D. Approximate Personalized Propagation of Neural Predictions (APPNPs)

Personalized propagation of neural predictions (PPNP) is constructed by using the propagation scheme based on the relationship between GCN and personalized PageRank [27], [28], [46], [47], [48]. Let $\mathbf{E}_0 = f(\mathbf{X}\mathbf{W}_0)$ be the mapped embedding with the learned weight $\mathbf{W}_0$, $\mathbf{I} \in \mathbb{R}^{n \times n}$ the identity matrix, ρ the predefined heuristic threshold, then the output of PPNP is $\mathbf{E} = \mathit{Softmax}\left(\rho\left(\mathbf{I} - (1-\rho)\,\mathbf{A}\right)^{-1}\mathbf{E}_0\right)$. In practice, to avoid the matrix inversion, Gasteiger et al. [28] designed APPNP to approximate PPNP in a power iteration manner, i.e.,

$$\mathbf{E}_\tau = \mathit{Softmax}\left((1-\rho)\,\mathbf{A}\mathbf{E}_{\tau-1} + \rho\mathbf{E}_0\right), \tag{3}$$

where τ is the number of power iteration steps. Notably, for GCN, the l-th feature $\mathbf{Z}_l$ could be formulated as $(\ldots\text{LReLU}(\mathbf{D}^{-\frac{1}{2}}(\mathbf{A} + \mathbf{I})\mathbf{D}^{-\frac{1}{2}}\mathbf{X}\mathbf{W}_0)\ldots\mathbf{W}_{l-1})$, where network parameters increase as the number of network layers increases. Differently, APPNP separates the mapping and propagation phases and utilizes the personalized PageRank to conduct messaging propagation, resulting in fewer parameters (only $\mathbf{W}_0$) and less training time. Although APPNP-based clustering approaches have significantly improved scalability, most of them have to pre-specify a heuristic threshold to determine the teleport probability, and few models focus on adaptively learning a suitable threshold to promote the representative capability in clustering tasks. Thus, we propose a simple yet effective strategy to initialize the threshold and learn it during the network training, namely IAPPNP. In this way, we utilize the IAPPNP modules to develop the scalable EGRC-Net. The framework comparison of typical deep embedding clustering approaches and our methods is illustrated in Figure 2.

III. Proposed Methods

In this section, we first present a novel embedding-induced graph refinement clustering network (EGRC-Net) framework, as shown in Figure 3. It mainly consists of the embedding learning module (Section III-A) and the graph refinement architecture (Section III-B). Then, we employ the Jeffreys

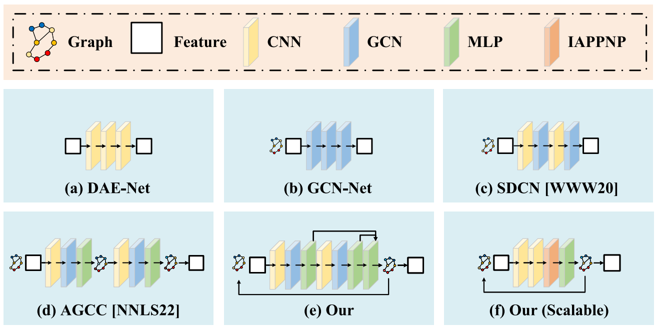


Fig. 2. Illustration of different network architectures for deep embedding clustering. (a) DAE-based clustering methods, (b) GCN-based clustering methods, (c) SDCN [19], (d) AGCC [21], (e) Our EGRC-Net and (f) Our scalable EGRC-Net. Colored cubes represent different network modules, where our proposed methods conduct the network training in an unsupervised manner to jointly conduct embedding learning and graph refinement, mutually benefiting both components.

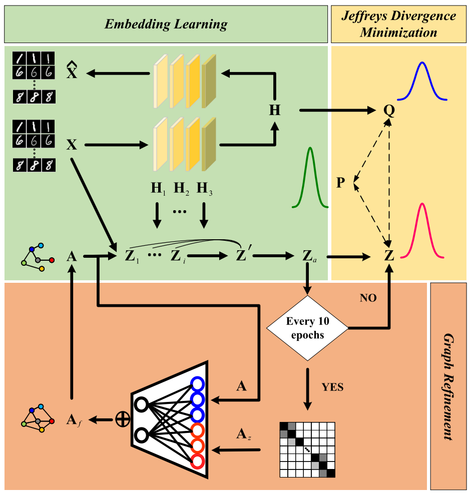


Fig. 3. The overall flowchart of the proposed method, namely embedding-induced graph refinement clustering network (EGRC-Net), where our proposed method conducts the network training in an unsupervised manner to jointly conduct embedding learning and graph refinement, mutually benefiting both components.

divergence minimization among multiple derived distributions to train the network in a unified optimization manner (Section III-C), capable of dynamically and jointly conducting embedding learning and graph refinement to mutually benefit both components. Moreover, we further designed an improved APPNP to replace the vanilla GCN for embedding learning, developing a simple yet effective scalable variant of EGRC-Net (Section III-D). Finally, the computational complexity of EGRC-Net and the scalable one is provided in Section III-E.

A. Embedding Learning

For embedding learning, we first utilize the vanilla DAE and GCN modules to extract the node attribute information and the topology structure information. Specifically, we use DAE to extract the i-th DAE feature representation $\mathbf{H}_i$ by Eq. (1) and the i-th GCN feature representation $\mathbf{Z}_i$ by Eq. (2). Then, we exploit a multilayer perceptron (MLP) layer with a normalization operation to learn the weight vectors of $\mathbf{H}_i$ and $\mathbf{Z}_i$ for subsequent information fusion, i.e.,

$$[\mathbf{m}_{i,1}, \mathbf{m}_{i,2}] = \ell_2\left(\mathit{Softmax}\left(\mathrm{LReLU}\left([\mathbf{H}_i \| \mathbf{Z}_i]\,\mathbf{W}_i^{\mathbf{F}}\right)\right)\right), \quad (4)$$

where $\mathbf{m}_{i,1}$ and $\mathbf{m}_{i,2}$ measure the importance of $\mathbf{H}_i$ and $\mathbf{Z}_i$, $\mathbf{W}_i^{\mathbf{F}} \in \mathbb{R}^{2d_i \times 2}$, $\cdot\|\cdot$, and ℓ_2 indicate a learnable weight matrix, the concatenation operation, and the ℓ_2 normalization, respectively. Afterward, based on the learned weight vectors, we can dynamically merge $\mathbf{H}_i$ and $\mathbf{Z}_i$, i.e.,

$$\mathbf{Z}_i' = \left(\mathbf{m}_{i,1}\mathbf{1}_i\right) \odot \mathbf{H}_i + \left(\mathbf{m}_{i,2}\mathbf{1}_i\right) \odot \mathbf{Z}_i, \quad (5)$$

where $\odot$ and $\mathbf{1}_i \in \mathbb{R}^{1\times d_i}$ indicate the Hadamard product of matrices and the all ones vector, respectively. Moreover, we aggregate the outputs of different GCN layers for fully utilizing the off-the-shelf information, where the weighted fusion process can be formulated as

$$\mathbf{Z}' = [(\mathbf{u}_1\mathbf{1}_1) \odot \mathbf{Z}_1 \| \cdots \| (\mathbf{u}_{l+1}\mathbf{1}_{l+1}) \odot \mathbf{Z}_{l+1}], \quad (6)$$

where we utilize the same weight learning strategy with a learnable weight matrix $\mathbf{W}^{\mathbf{S}}$ to learn the corresponding weights of multi-scale features from different GCN layers. The formulation is as follows, $\mathbf{U} = \ell_2(\mathit{Softmax}(\mathrm{LReLU}([\mathbf{Z}_1 \| \cdots \| \mathbf{Z}_{l+1}]\mathbf{W}^{\mathbf{S}})))$, where $\mathbf{u}_i$ is the i-th element of $\mathbf{U}$ and $l = 4$ in the experiments. Notably, we use a weighted concatenation form to conduct feature fusion because the outputs of different GCN layers have different dimensions. In addition, we use the Laplacian smoothing technique, a learnable weight matrix $\mathbf{W}^{\mathbf{Z}}$, and the Softmax function to learn the final embedding, i.e.,

$$\mathbf{Z}_a = \mathit{Softmax}(\mathbf{D}^{-\frac{1}{2}}(\mathbf{A}+\mathbf{I})\mathbf{D}^{-\frac{1}{2}}\mathbf{Z}'\mathbf{W}^{\mathbf{Z}}), \quad (7)$$

where $\mathbf{Z}_a \in \mathbb{R}^{n\times\kappa}$ indicates the learned embedding with κ being the cluster number.

B. Graph Refinement

The adjacency matrix $\mathbf{A}$ is crucial in GCN-based graph clustering. However, earlier works heavily relied on the adjacency matrix of a predefined graph and may fail if that fixed graph cannot well reflect the intrinsic semantic structure of the embedding space. Such a phenomenon inspires us to propose a novel graph clustering network that adaptively refines the graph rather than relies on the fixed one.

For graph refinement, we construct a local neighborhood graph to approximate the manifold construction in the embedding space, preserving the local geometry of neighborhoods. Specifically, we first employ the extracted feature representation $\mathbf{Z}_a$ and its similarity matrix $\mathbf{S}_z = \left(\mathbf{Z}_a\mathbf{Z}_a^{\top}\right) / \left(\|\mathbf{Z}_a\|_F \left\|\mathbf{Z}_a^{\top}\right\|_F\right)$ to construct a graph $\mathbf{G}$ with its element

$$g_{i,j} = \begin{cases} s_{i,j} & \text{if } s_{i,j} == \max\left(\mathbf{S}_i\right), \\ 0 & \text{otherwise,} \end{cases}$$
$$\text{s.t.} \quad \mathbf{S} = \mathbf{S}_z - \mathrm{diag}\left(\mathbf{S}_z\right), \quad (8)$$

where $s_{i,j}$ is the (i, j)-th element of $\mathbf{S}$, $\max\left(\mathbf{S}_i\right)$ is the maximum value of $\mathbf{S}$ in the i-th row, $\mathrm{diag}\left(\mathbf{S}_z\right)$ is a diagonal matrix containing the elements of the column vector of the main diagonal elements of $\mathbf{S}_z$. Here, we use cosine similarity because it is not affected by the absolute magnitude of the node vector, which is important to indicate the similarity of the feature representation. Such a graph construction way exclusively focuses on the nearest neighbor of samples, which is expected to be the most reliable information. Then, since a graph neural network can propagate information more reliably when self-loops are considered [49], we apply the self-loop operation to normalize $\mathbf{G}$, and then symmetrize it, i.e.,

$$g_{i,j} = g_{j,i} = \begin{cases} \max\{g_{i,j}, g_{j,i}\} & \text{if } i \neq j, \\ 1 & \text{if } i == j, \end{cases} \tag{9}$$

where $\max\{g_{i,j}, g_{j,i}\}$ denotes the maximum value among $g_{i,j}$ and $g_{j,i}$, and we can obtain the adjacency matrix $\mathbf{A}_z = \mathbf{D}_z^{-1}\mathbf{G}$ with its degree matrix $\mathbf{D}_z$. Then, we build a multilayer perceptron layer parametrized by a weight matrix $\mathbf{W}^{\mathbf{A}} \in \mathbb{R}^{2n\times 2}$ to capture the relationship among the adjacency matrix $\mathbf{A}$ of the initial graph and the constructed adjacency matrix $\mathbf{A}_z$ with a normalization operation, which can be formulated as

$$\mathbf{V} = [\mathbf{v}_z \| \mathbf{v}] = \ell_2\left(Softmax\left(\text{LReLU}\left(\left[\mathbf{A}_z \| \mathbf{A}\right]\mathbf{W}^{\mathbf{A}}\right)\right)\right), \tag{10}$$

where $\mathbf{V} \in \mathbb{R}^{n\times 2}$ is the weight coefficient matrix with entries being greater than 0, $\mathbf{v}_z$ and $\mathbf{v}$ are the weight vectors for measuring the importance of $\mathbf{A}_z$ and $\mathbf{A}$, respectively. Thus, we can fuse $\mathbf{A}_z$ and $\mathbf{A}$ in an adaptive fusion manner as

$$\mathbf{A}_f = (\mathbf{v}_z\mathbf{1}) \odot \mathbf{A}_z + (\mathbf{v}\mathbf{1}) \odot \mathbf{A}, \tag{11}$$

where $\mathbf{1} \in \mathbb{R}^{1\times n}$ denotes the vector of all ones. Finally, we iteratively perform graph refinement to boost the adjacency matrix of the graph, promoting graph embedding learning. In each iteration, only the nearest neighbor connection information is added to the graph fusion process. In addition, as the number of iterations increases, more information is incorporated into the adjacency matrix. The adjacency matrix is updated every i_p epoch, with $i_p = 10$ being used empirically. More experiments and analyses are given in Section IV-F. 2.

C. Clustering Optimization

Since clustering is an unsupervised task, we have designed a practical end-to-end optimization method to train the network in a unified optimization manner. We start by calculating the similarity between an embedding $\mathbf{h}_i$ and its centroid $\boldsymbol{\mu}_j$ using the Student's t-distribution function [50], [51]. This yields a similarity that can be represented as a probability distribution $\mathbf{Q}$ and we can formulate its (i, j)-th element as

$$q_{i,j} = \frac{(1 + \|\mathbf{h}_i - \boldsymbol{\mu}_j\|^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_{j'}(1 + \|\mathbf{h}_i - \boldsymbol{\mu}_{j'}\|^2/\alpha)^{-\frac{\alpha+1}{2}}}, \tag{12}$$

where $\mathbf{H} = \mathbf{H}_l = [\mathbf{h}_1, \cdots, \mathbf{h}_n]^{\mathsf{T}}$, $\boldsymbol{\mu}_j$ is obtained with K-means [24] based on the feature $\mathbf{h}_j$, and α is set to 1. Then, we derive an auxiliary distribution from the element of $\mathbf{Z}_a$ (i.e., $z_{i,j}$) following [15], i.e.,

$$p_{i,j} = \frac{z_{i,j}^2/\sum_i z_{i,j}}{\sum_{j'} z_{i,j'}^2/\sum_i z_{i,j'}}, \tag{13}$$

where $1 \geq p_{i,j} \geq 0$ is the element of the auxiliary distribution $\mathbf{P}$. Finally, to achieve the consistent alignment of multiple derived distributions, we minimize the Jeffreys divergence between $\mathbf{Q}$, $\mathbf{Z}_a$, and $\mathbf{P}$. By combining with Eq. (1), the loss function of our method can be written as

$$\begin{aligned}\min_{\mathbf{Z}_a} \left\|\mathbf{X} - \hat{\mathbf{X}}\right\|_F^2 &+ \lambda_1 \sum_i^n \sum_j^\kappa \left(p_{i,j} log\frac{p_{i,j}}{z_{i,j}} + z_{i,j} log\frac{z_{i,j}}{p_{i,j}}\right) \\ &+ \lambda_2 \sum_i^n \sum_j^\kappa \left(p_{i,j} log\frac{p_{i,j}}{q_{i,j}} + q_{i,j} log\frac{q_{i,j}}{p_{i,j}}\right) \\ &+ \lambda_3 \sum_i^n \sum_j^\kappa \left(z_{i,j} log\frac{z_{i,j}}{q_{i,j}} + \quad q_{i,j} log\frac{q_{i,j}}{z_{i,j}}\right),\end{aligned} \tag{14}$$

where $\lambda_1, \lambda_2, \lambda_3 > 0$ are the trade-off parameters. When the network training is well accomplished, $\mathbf{Z}_a$ can directly indicate the clustering assignments as

$$\begin{aligned} y_i &= \arg\max_j z_{i,j} \\ \text{s.t.} \quad & j = 1, \cdots, \kappa, \end{aligned} \tag{15}$$

where y_i is the inferred prediction with respect to (w.r.t.) the data $\mathbf{x}_i$. The training process of our method EGRC-Net is summarized in Alg. 1.

D. Scalable EGRC-Net

We further develop EGRC-Net in a scalable manner to explore its scalability to handle the practical large-graph problem [23], [25], [26]. It is well known that the scalability of GCN-based clustering methods has been limited due to the utilization of GCN since its explicit message-passing leads to an expensive neighborhood expansion [27]. Thus, we early utilize the approximate personalized propagation of neural predictions (APPNP) [28] instead of multi-layer graph convolution to complete the transmission and aggregation of node information on the graph. However, we observe that APPNP requires pre-specifying a heuristic threshold that determines the teleport probability. Besides, its value limits its availability in real-world applications since there is a little guideline to set a suitable threshold in clustering tasks. To this end, we empirically use a random number generator to sample from a uniform distribution on the interval [0, 1). Then, we set it as trainable and calculate its gradient with the backpropagation. Afterward, we can achieve the information propagation by using the personalized PageRank with a learned teleport probability value rather than relying on predefined heuristic thresholds. By imposing the designed IAPPNP module, the graph embedding learning of the scalable EGRC-Net formulation can be written as

$$\begin{aligned} \mathbf{Z}_a &= \mathbf{E}_\tau \quad \text{s.t.} \quad \iota = 1, \ldots, \tau, \\ \mathbf{E}_\iota &= Softmax\left((1 - \Theta)\,\mathbf{E}_0 + \Theta\mathbf{A}\mathbf{E}_{\iota-1}\right), \end{aligned} \tag{16}$$

where $\mathbf{Z}_a$ is the final output of IAPPNP, Θ is the learned threshold, $\mathbf{E}_0$ is the linear mapping of $\mathbf{X}$ via MLP. We uniformly set $\tau = 1$ here, and more specific analyses are given in Section IV-J.

Algorithm 1 Training Process of the Proposed EGRC-Net

```
Input: Input data $\mathbf{X}$; Adjacency matrix $\mathbf{A}$; Parameters $\lambda_1, \lambda_2, \lambda_3$; Cluster number $\kappa$;
Output: Clustering assignments $\mathbf{y}$;
1:  Let $i_{Iter} = 1$, $i_{all} = 200$, $i_p = 10$;
2:  Initialize the parameters of DAE;
3:  while $i_{Iter} < i_{all}$ do
4:      Calculate the matrix $\mathbf{H}$ by Eq. (1);
5:      Calculate the matrix $\mathbf{Z}_a$ by Eq. (7);
6:      if $i_{Iter} \% i_p == 0$ then
7:          Update the adjacency matrix $\mathbf{A}$ by Eq. (11);
8:      end if
9:      Calculate the distribution $\mathbf{Q}$ by Eq. (12);
10:     Calculate the distribution $\mathbf{P}$ by Eq. (13);
11:     Calculate the overall loss function by Eq. (14);
12:     Initialize the parameter gradient of the network to zero; perform the backpropagation; update all parameters of the network;
13:     $i_{Iter} = i_{Iter} + 1$;
14: end while
15: Obtain the final clustering assignments $\mathbf{y}$ by Eq. (15);
```

E. Computational Complexity

For DAE, the computational complexity is $\mathcal{O}(n\sum_{i=2}^{l} d_{i-1}d_i)$; for GCN, the computational complexity is $\mathcal{O}(|\mathcal{E}|\sum_{i=2}^{l} d_{i-1}d_i)$ corresponding to [52], where $|\mathcal{E}|$ is the number of edges; for IAPPNP, the computational complexity is $\mathcal{O}(|\mathcal{E}|d\kappa)$; for Eq. (12), the computational complexity is $\mathcal{O}(n\kappa + n\log n)$ corresponding to [15]; for the MLP modules, the computational complexity is $\mathcal{O}(\sum_{i=1}^{l-1}(d_i) + (\sum_{i=1}^{l+1} d_i)(l+1))$; for the graph refinement module, the computational complexity is n. Thus, the total computational complexity of EGRC-Net in one iteration is $\mathcal{O}(n\sum_{i=2}^{l} d_{i-1}d_i + |\mathcal{E}|\sum_{i=2}^{l} d_{i-1}d_i + n\kappa + n\log n + \sum_{i=1}^{l+1} d_i + l\sum_{i=1}^{l+1} d_i + n)$, while that of the scalable one is $\mathcal{O}(n\sum_{i=2}^{l} d_{i-1}d_i + |\mathcal{E}|d\kappa + n\kappa + n\log n + \sum_{i=1}^{l+1} d_i + l\sum_{i=1}^{l+1} d_i + n)$.

IV. Experiments

In this section, we first present the used benchmark datasets (Section IV-A), the compared methods (Section IV-B), the evaluation metrics (Section IV-C), and the network implementation details (Section IV-D). Then, we conduct quantitative experiments and analyses (Section IV-E) to evaluate the effectiveness of our method EGRC-Net. Afterward, we further investigate EGRC-Net by conducting parameters analyses (Section IV-F), model stability analysis (Section IV-G), ablation studies (Section IV-H), and visual comparisons in Section IV-I. Furthermore, we design the experiments w.r.t. the different strategies of the teleport probability value and the number of propagation steps to investigate the effectiveness of the scalable EGRC-Net (Section IV-J). Finally, we compare our methods and the state-of-the-art approaches w.r.t. running time and network parameters (Section IV-K).

TABLE II
Summary of the Used Datasets

Dataset	Type	Samples	Dimension	Classes	Edges
USPS	Image	9298	256	10	27894
STL10	Image	13000	512	10	39000
ImageNet-10	Image	13000	512	10	39000
HHAR	Record	10299	561	6	30897
REUT	Text	10000	2000	4	30000
ACM	Graph	3025	1870	3	13128
CITE	Graph	3327	3703	6	4552
DBLP	Graph	4057	334	4	3528
PubMed	Graph	19717	500	3	44324

A. Datasets

We conducted the comparisons on nine commonly used datasets, including USPS [53], STL10 [54], ImageNet-10 [55], HHAR [56]), REUT [57], ACM[1], CITE[2], DBLP[3], and PubMed [58], which are summarized in Table II. Particularly, We adopt ResNet34 [59] to preprocess STL10 and ImageNet-10, where its output dimension of the fully-connected classification layer is set to 512. The corresponding details are provided as follows.

- **USPS**. The USPS dataset is the earliest version of USPS, which has ten classes, labeled '0' to '9'. It comprises 9,298 handwritten digit images, all of which have been standardized to a uniform size 16 × 16.
- **STL10**. The STL10 dataset is an image dataset derived from ImageNet and popularly used to evaluate algorithms of unsupervised feature learning. It contains 13,000 labeled images from 10 object classes (such as birds, cats, trucks).
- **ImageNet-10**. The ImageNet-10 dataset is a small-scale subset of the ImageNet database, containing 13,000 labeled images from 10 object classes. Although significantly smaller, it retains the structure and diversity of the original ImageNet dataset.
- **HHAR**. The HHAR dataset encompasses 10,299 sensor readings collected from smartphones and smartwatches. These readings have been divided into six categories that correspond to different human activities, including biking, sitting, standing, walking, stair up, and stair down.
- **REUT**. The REUT dataset comprises a compilation of English news articles, organized into categorical labels. Due to scalability limitations of certain algorithms with the full Reuters (REUT) dataset (685,071 samples), we employ a widely used smaller version (10,000 samples) of the REUT dataset for comparison purposes [15].
- **ACM**. The ACM dataset is a network of scholarly papers sourced from the ACM digital library, where two papers are linked with an edge if they share a common author. The features of the papers were derived from keywords associated with the KDD, SIGMOD, SIGCOMM, and MobiCOMM conferences, and were categorized into three classes based on their research area: database, wireless communication, and data mining.

[1]http://dl.acm.org
[2]http://CiteSeerx.ist.psu.edu/
[3]https://dblp.uni-trier.de

- **CITE**. The CITE dataset is a network of citations comprised of sparse bag-of-words feature vectors for each document and a collection of citation links between documents. The labels for the documents are organized into six distinct research areas: agents, artificial intelligence, database, information retrieval, machine language, and human-computer interaction.
- **DBLP**. The DBLP dataset is a network of authors sourced from the DBLP computer science bibliography, where two authors are linked with an edge if they have a co-author relationship. The author's features are represented as elements in a bag-of-words, constructed from keywords, and the authors are divided into four research areas: database, data mining, machine learning, and information retrieval. The authors are labeled based on the conference they have submitted to.
- **PubMed**. The PubMed dataset comprises 19,717 scientific publications related to diabetes, classified into one of three categories, sourced from the PubMed database. The citation network within the dataset includes 44,324 links. Each publication is represented by a Term Frequency/Inverse Document Frequency weighted word vector, derived from a dictionary of 500 unique words.

B. Compared Methods

The compared methods include K-means [24], three DAE-based clustering methods (DAE [14], DEC [15], and IDEC [16]), eight GCN-based clustering methods (GAE [17], VGAE [17], ARGA [22], DAEGC [18], FastGAE [41], RG-VGAE [42], MA-GAE [43], and EGAE [23]), as well as three DAE and GCN combination-based clustering methods (SDCN [19], AGCN [20], and AGCC [21]).

C. Evaluation Metrics

The performance of the clustering was quantitatively evaluated using three commonly used metrics: adjusted rand index (ARI), accuracy (ACC), and normalized mutual information (NMI). The larger values of these metrics indicate a higher quality of clustering.

D. Implementation Details

For fair comparisons, the dimensions of the vanilla DAE and GCN layers were set to 500-500-2000-10, as in previous studies [15], [19], [20], [21]. The DAE module was first pre-trained for 30 epochs using a learning rate of 0.001. The entire network was then fine-tuned with $i_{all} = 200$. The learning rate was set to 0.001 for the USPS, HHAR, ACM, DBLP, and PubMed datasets and 0.0001 for the REUT and CITE datasets. For the approaches FastGAE, RG-VGAE, MA-GAE, EGAE, and AGCC, we used their publicly available codes and parameter settings given by the original papers [21], [23], [41], [42], [43]. For other methods under comparison, we directly cited the results in [45]. The results were reported as the mean values and their corresponding standard deviations, obtained by repeating the experiments ten times, represented as mean $\pm$ std in the experimental results. The training was implemented using PyTorch on a GeForce RTX 2080 Ti, a GeForce RTX 3090, and a Quadro RTX 8000 GPU.

E. Quantitative Results

Table III provides the quantitative comparisons of the proposed method EGRC-Net and fifteen compared approaches on nine benchmark datasets w.r.t. three metrics, where we can draw the following conclusions.

- EGRC-Net obtains the best experimental results in all metrics on nine benchmark datasets. For instance, in **non-graph** dataset USPS, our EGRC-Net enhances the ARI, ACC, and NMI metrics of the second-best approach AGCN [20] by 2.00%, 2.43%, and 1.30% respectively. In the **graph** dataset CITE, our EGRC-Net improves 4.53% over AGCN on ARI, 3.48% on ACC, and 4.23% on NMI.
- DAEGC [18] outperforms GAE [17] in almost all metrics on clustering performance, validating the effectiveness of exploiting an adaptive fusion mechanism. In this paper, we develop the embedding learning module and the graph refinement architecture in an adaptive fusion manner to flexibly explore the multiple off-the-shelf information, boosting graph clustering performance.
- AGCC [21] obtains the second-best results on the commonly used graph dataset DBLP; however, it suffers from the out-of-memory case on the larger graph PubMed. Differently, our method EGRC-Net can get the best results on PubMed at the cost of acceptable resource consumption. Such a phenomenon also emphasizes the importance of network scalability.
- EGRC-Net obtains a significant improvement on DBLP, e.g., we enhance the ARI, ACC, and NMI metrics of the best-compared approach AGCC [21] by 11.99%, 7.08%, and 10.49% respectively. The reason may be attributed to the fact that DBLP has low feature dimensions, i.e., little learnable information, meaning that sufficiently utilizing the available off-the-shelf embedding and graph information greatly improves the clustering performance.

F. Parameters Analysis

1) Analysis of Hyperparameters: Our loss function (i.e., Eq. (14)) has three hyperparameters (λ_1, λ_2, and λ_3) to balance the contributions of multiple loss terms. We conducted the parameter analysis in Figure 4, where we can find that λ_1 and λ_2 (w.r.t. the distribution $\mathbf{P}$) should be larger than λ_3, reflecting the importance of $\mathbf{P}$ in guiding the network training. In addition, we observe that λ_1 (w.r.t. $\mathbf{Z}_a$) should be larger than λ_2 because $\mathbf{Z}_a$ holds the node attribute and topology structure information by integrating the deep auto-encoder and graph convolution network features, resulting in a stronger guidance ability.

2) Analysis of i_p*:* We investigated the effect of the epoch interval of graph refinement i_p on DBLP and ACM in Figure 5, where we have the following conclusions.

- Overmuch frequency of graph refinement (e.g., $i_p = 1$) leads to high time cost and can lead to performance degradation. The possible reason is that the early training does not learn a useful embedding, and the subsequent graph refinement will cause the model to fall into a suboptimal solution.

TABLE III

EXPERIMENTAL COMPARISONS. WE HIGHLIGHTED THE BEST AND SECOND-BEST RESULTS WITH **BOLD** AND UNDERLINE, RESPECTIVELY. 'OOM' DENOTES THE OUT-OF-MEMORY ISSUE

Datasets	Metrics	K-means [24]	DAE [14] [Science06]	DEC [15] [ICML16]	IDEC [16] [AAAI17]	GAE [17] [NIPS16]	VGAE [17] [NIPS16]	ARGA [22] [IJCAI18]	DAEGC [18] [AAAI19]
USPS	ARI	54.55±0.06	58.83±0.05	63.70±0.27	67.86±0.12	50.30±0.55	40.96±0.59	51.10±0.60	63.33±0.34
	ACC	66.82±0.04	71.04±0.03	73.31±0.17	76.22±0.12	63.10±0.33	56.19±0.72	66.80±0.70	73.55±0.40
	NMI	62.63±0.05	67.53±0.03	70.58±0.25	75.56±0.06	60.69±0.58	51.08±0.37	61.60±0.30	71.12±0.24
STL10	ARI	67.24±0.12	69.13±0.17	08.08±1.13	36.76±4.51	70.6±1.38	66.95±2.93	45.30±6.02	84.03±0.85
	ACC	83.53±0.07	84.36±0.11	28.34±1.10	53.95±2.93	82.82±0.51	79.65±1.97	68.28±4.08	92.22±0.42
	NMI	75.43±0.03	76.18±0.11	13.81±1.53	50.78±4.53	82.43±0.76	79.25±1.88	58.35±5.11	84.46±0.64
ImageNet-10	ARI	69.03±0.05	69.73±0.62	03.00±0.49	03.24±1.22	56.78±4.98	72.74±1.47	25.48±5.08	11.00±1.21
	ACC	76.70±0.02	77.36±1.33	19.61±0.79	18.36±1.21	73.66±3.22	83.10±0.51	47.67±6.81	30.68±1.85
	NMI	75.89±0.04	76.44±0.36	05.63±0.90	05.38±1.75	66.00±2.83	81.50±2.22	34.86±4.21	17.71±1.38
HHAR	ARI	46.09±0.02	60.36±0.88	61.25±0.51	62.83±0.45	42.63±1.63	51.47±0.73	44.70±1.00	60.38±2.15
	ACC	59.98±0.02	68.69±0.31	69.39±0.25	71.05±0.36	62.33±1.01	71.30±0.36	63.30±0.80	76.51±2.19
	NMI	58.86±0.01	71.42±0.97	72.91±0.39	74.19±0.39	55.06±1.39	62.95±0.36	57.10±1.40	69.10±2.28
REUT	ARI	27.95±0.38	49.55±0.37	48.44±0.14	51.26±0.21	19.61±0.22	26.18±0.36	24.50±0.40	31.12±0.18
	ACC	54.04±0.01	74.90±0.21	73.58±0.13	75.43±0.14	54.40±0.27	60.85±0.23	56.20±0.20	65.50±0.13
	NMI	41.54±0.51	49.69±0.29	47.50±0.34	50.28±0.17	25.92±0.41	25.51±0.22	28.70±0.30	30.55±0.29
ACM	ARI	30.60±0.69	54.64±0.16	60.64±1.87	62.16±1.50	59.46±3.10	57.72±0.67	62.90±2.10	59.35±3.89
	ACC	67.31±0.71	81.83±0.08	84.33±0.76	85.12±0.52	84.52±1.44	84.13±0.22	86.10±1.20	86.94±2.83
	NMI	32.44±0.46	49.30±0.16	54.54±1.51	56.61±1.16	55.38±1.92	53.20±0.52	55.70±1.40	56.18±4.15
CITE	ARI	13.43±3.02	29.31±0.14	28.12±0.36	25.70±2.65	33.55±1.18	33.13±0.53	33.40±1.50	37.78±1.24
	ACC	39.32±3.17	57.08±0.13	55.89±0.20	60.49±1.42	61.35±0.80	60.97±0.36	56.90±0.70	64.54±1.39
	NMI	16.94±3.22	27.64±0.08	28.34±0.30	27.17±2.40	34.63±0.65	32.69±0.27	34.50±0.80	36.41±0.86
DBLP	ARI	06.97±0.39	12.21±0.43	23.92±0.39	25.37±0.60	22.02±1.40	17.92±0.07	22.70±0.30	21.03±0.52
	ACC	38.65±0.65	51.43±0.35	58.16±0.56	60.31±0.62	61.21±1.22	58.59±0.06	61.60±1.00	62.05±0.48
	NMI	11.45±0.38	25.40±0.16	29.51±0.28	31.17±0.50	30.80±0.91	26.92±0.06	26.80±1.00	32.49±0.45
PubMed	ARI	28.10±0.01	23.86±0.67	19.55±0.13	20.58±0.39	20.62±1.39	30.15±1.23	24.35±0.17	29.84±0.04
	ACC	59.83±0.01	63.07±0.31	60.14±0.09	60.70±0.34	62.09±0.81	68.48±0.77	65.26±0.12	68.73±0.03
	NMI	31.05±0.02	26.32±0.57	22.44±0.14	23.67±0.29	23.84±3.54	30.61±1.71	24.80±0.17	28.26±0.03

Datasets	Metrics	SDCN [19] [WWW20]	AGCN [20] [MM21]	FastGAE [41] [NN21]	RG-VGAE [42] [TKDE22]	MA-GAE [43] [NN22]	EGAE [23] [NNLS22]	AGCC [21] [NNLS22]	Our
USPS	ARI	71.84±0.24	73.61±0.43	00.36±0.05	04.82±0.07	03.09±0.17	64.13±2.22	68.50±3.83	**75.61±1.92**
	ACC	78.08±0.19	80.98±0.28	13.34±0.28	22.42±0.11	31.33±0.69	76.22±2.48	77.14±1.21	**83.41±4.40**
	NMI	79.51±0.27	79.64±0.32	00.93±0.06	12.22±0.12	26.72±1.03	70.83±2.25	75.93±3.83	**80.94±0.92**
STL10	ARI	84.09±0.72	84.59±0.26	10.80±1.25	01.48±0.02	04.07±0.89	85.73±0.14	OOM	**85.92±0.04**
	ACC	92.27±0.38	92.57±0.14	42.45±3.08	17.08±0.08	26.45±1.41	93.13±0.07	OOM	**93.23±0.02**
	NMI	85.03±0.42	85.27±0.15	35.68±3.76	05.93±0.05	22.74±0.74	85.98±0.14	OOM	**86.07±0.02**
ImageNet-10	ARI	72.41±0.34	73.38±0.73	06.04±1.90	05.29±0.15	04.01±0.64	69.84±4.25	OOM	**78.38±3.62**
	ACC	78.37±0.25	79.30±0.29	25.91±2.52	22.14±0.35	31.57±2.10	84.66±2.62	OOM	**87.29±3.90**
	NMI	79.83±0.26	80.58±0.73	10.26±2.59	10.04±0.06	22.28±2.89	72.18±3.26	OOM	**82.28±1.80**
HHAR	ARI	72.84±0.09	77.07±0.66	00.07±0.05	16.29±0.06	15.68±1.55	59.37±2.76	75.58±1.85	**78.22±0.30**
	ACC	84.26±0.17	88.11±0.43	19.39±0.30	40.11±0.07	49.50±1.22	75.97±1.05	86.54±1.79	**88.95±0.21**
	NMI	79.90±0.09	82.44±0.62	00.45±0.11	27.51±0.10	37.59±2.41	67.26±2.30	82.21±1.78	**82.48±0.30**
REUT	ARI	55.36±0.37	60.55±1.78	01.12±0.66	05.86±0.31	02.84±4.21	48.12±3.90	62.98±2.24	**63.53±0.66**
	ACC	77.15±0.21	79.30±1.07	40.11±0.64	43.06±0.38	40.69±5.13	72.12±2.67	81.65±1.52	**81.90±0.16**
	NMI	50.82±0.21	57.83±1.01	00.39±0.13	04.39±0.08	07.08±4.42	49.17±3.75	59.56±0.94	**60.32±0.40**
ACM	ARI	73.91±0.40	74.20±0.38	03.08±0.55	18.48±0.30	63.89±1.33	73.97±1.01	73.73±0.90	**76.04±0.39**
	ACC	90.45±0.18	90.59±0.15	43.25±1.45	63.85±0.23	86.45±0.56	90.51±0.41	90.38±0.38	**91.30±0.17**
	NMI	68.31±0.25	68.38±0.45	13.16±2.34	32.58±0.25	57.31±1.31	67.71±0.92	68.34±0.89	**70.40±0.30**
CITE	ARI	40.17±0.43	43.79±0.31	03.91±2.07	37.65±0.41	28.73±2.89	38.72±1.00	41.82±2.03	**48.32±0.57**
	ACC	65.96±0.31	68.79±0.23	28.31±2.62	61.75±0.39	55.41±3.10	64.38±0.81	68.08±1.44	**72.27±0.37**
	NMI	38.71±0.32	41.54±0.30	05.61±3.32	38.27±0.32	30.52±1.72	38.34±0.98	40.86±1.45	**45.77±0.48**
DBLP	ARI	39.15±2.01	42.49±0.31	02.42±0.46	08.03±0.40	06.17±1.64	31.64±2.34	44.40±3.79	**56.39±0.93**
	ACC	68.05±1.81	73.26±0.37	33.85±1.04	42.61±0.38	40.43±1.31	63.24±2.17	73.45±2.16	**80.53±0.54**
	NMI	39.50±1.34	39.68±0.42	03.33±1.05	11.76±0.32	11.36±1.37	31.26±2.42	40.36±2.81	**50.85±0.60**
PubMed	ARI	22.30±2.07	31.39±0.67	12.08±6.58	07.30±0.13	30.55±2.82	33.00±1.74	OOM	**34.45±0.58**
	ACC	64.20±1.30	69.67±0.42	53.01±5.45	52.51±0.58	68.82±1.57	70.69±1.25	OOM	**71.91±0.22**
	NMI	22.87±2.04	30.96±0.99	18.09±4.34	07.17±0.33	29.86±2.76	31.54±1.95	OOM	**32.15±1.81**

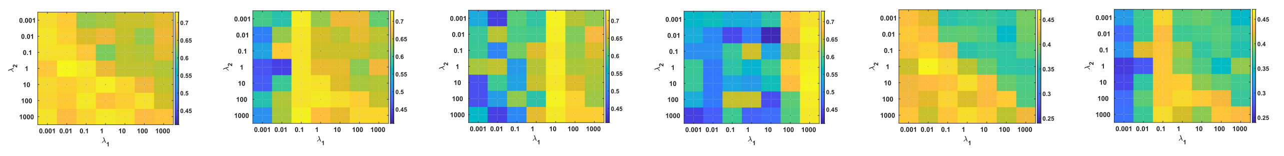
- Setting i_p to an appropriate value is capable of reducing numerous time consumption with a good clustering performance, e.g., $i_p = 10$. Therefore, we empirically set $i_p = 10$ in the experiments, which obtains a significant clustering improvement at the cost of acceptable time consumption.

G. Model Stability Analysis

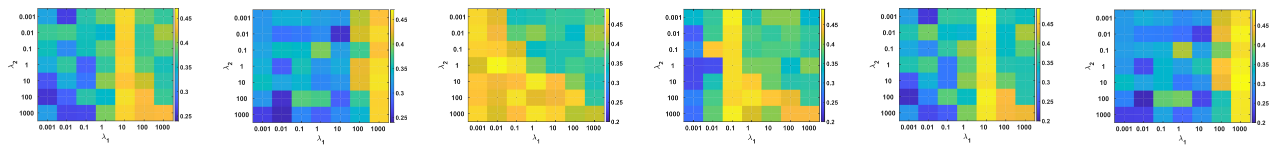
We conducted experiments on REUT to verify the training stability of the proposed EGRC-Net model. The overall loss and accuracy values concerning different epochs are shown in Figure 6, where we first find that the loss function value gradually becomes stable after 150 iterations, where the proposed network can effectively converge within 200 iterations. In addition, we plotted 2D t-distributed stochastic neighbor embedding (t-SNE) visualizations [60] of the learned embeddings w.r.t. the epoch number in Figure 7. Since there is no globally-accepted metric in the existing literature for the t-SNE visualization, we used the homogeneity and completeness scores [11], [61] to quantitatively measure the visualization performance. A visualization result satisfies the high homogeneity if all of its groups contain only data points that are members of a single class, and that one satisfies the high completeness if all the data points that are members of a given class are elements of the same cluster. These metrics quantitatively measure the visualized quality of the embeddings with a higher score indicating a better representation. From Figures 6 (b) and 7, we observe that, as the epoch increases, the learned embeddings gradually become better, and the corresponding clustering accuracy also becomes higher, demonstrating that the network training is stable and effective.

H. Ablation Study

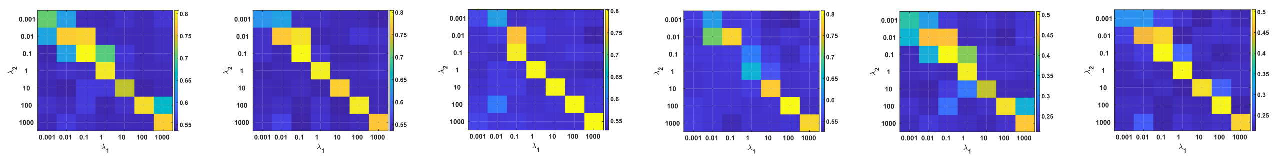
We conducted a series of ablation experiments to evaluate the effectiveness of the proposed graph refinement (GR) architecture and the Jeffreys divergence (JD) minimization term, where the experimental results are shown in Table IV, ✗and ✓in each row indicate the non-use and use of the corresponding component, respectively. Specifically, non-use of the GR term means that we use a fixed graph to conduct graph clustering, and non-use of the JD term means that we minimize an asymmetric divergence between $\mathbf{Q}$, $\mathbf{Z}_a$, and $\mathbf{P}$ (i.e., $\lambda_1 \sum_i^n \sum_j^\kappa p_{i,j} log \frac{p_{i,j}}{z_{i,j}} + \lambda_2 \sum_i^n \sum_j^\kappa p_{i,j} log \frac{p_{i,j}}{q_{i,j}} + \lambda_3 \sum_i^n \sum_j^\kappa z_{i,j} log \frac{z_{i,j}}{q_{i,j}}$). As shown in Table IV, we can find that the GR architecture or the JD minimization term improves the clustering performance on most benchmark datasets, verified by comparing the first and second-row results and the first



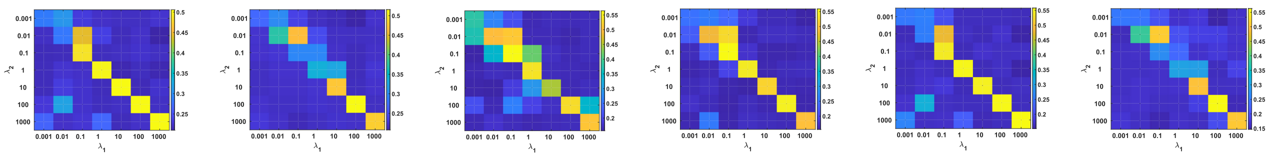
(a) λ_3=0.001 (ACC) (b) λ_3=0.1 (ACC) (c) λ_3=10 (ACC) (d) λ_3=1000 (ACC) (e) λ_3=0.001 (NMI) (f) λ_3=0.1 (NMI)



(g) λ_3=10 (NMI) (h) λ_3=1000 (NMI) (i) λ_3=0.001 (ARI) (j) λ_3=0.1 (ARI) (k) λ_3=10 (ARI) (l) λ_3=1000 (ARI)

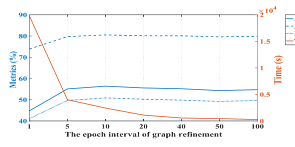


(m) λ_3=0.001 (ACC) (n) λ_3=0.1 (ACC) (o) λ_3=10 (ACC) (p) λ_3=1000 (ACC) (q) λ_3=0.001 (NMI) (r) λ_3=0.1 (NMI)

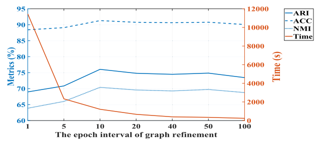


(s) λ_3=10 (NMI) (t) λ_3=1000 (NMI) (u) λ_3=0.001 (ARI) (v) λ_3=0.1 (ARI) (w) λ_3=10 (ARI) (x) λ_3=1000 (ARI)

Fig. 4. Analysis of different hyperparameters (λ_1, λ_2, and λ_3) with three metrics on CITE (i.e., (a) - (l)) and DBLP (i.e., (m) - (x)). The results of these hyperparameters are visually depicted in a 3D figure, where the color represents the third dimension, i.e., the experimental outcomes.

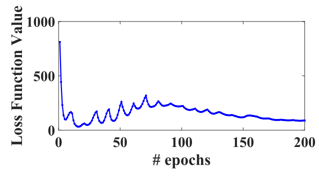


(a) DBLP

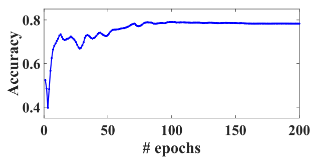


(b) ACM

Fig. 5. Parameter analysis w.r.t. the epoch interval of graph refinement on (a) DBLP and (b) ACM.



(a) Overall Loss



(b) Accuracy

Fig. 6. Training stability analysis w.r.t. (a) the loss value and (b) accuracy with different epochs on REUT.

and third-row results in all metrics. In addition, comparing the second (using the fixed graph based on JD) and fourth (using our refined graph based on JD) row results, we can find that the GR strategy can achieve the best clustering performance among all metrics based on the JD minimization, validating the effectiveness of the proposed method.

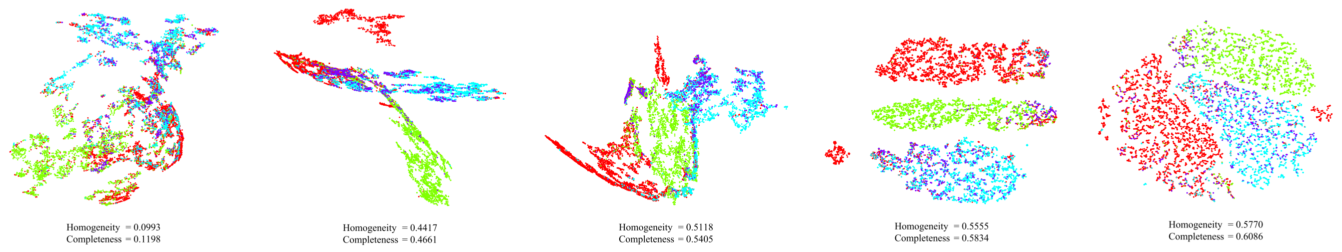
TABLE IV

THE ABLATION STUDY OF THE PROPOSED GRAPH REFINEMENT (GR) ARCHITECTURE AND THE JEFFREYS DIVERGENCE (JD) MINIMIZATION TERM, WHERE ✗ AND ✓ IN EACH ROW INDICATE THE NON-USE OR USE OF THE CORRESPONDING COMPONENT, RESPECTIVELY. WE HIGHLIGHTED THE BEST RESULTS WITH **BOLD**

	GR	JD	ARI	ACC	NMI
USPS	✗	✗	70.81±1.03	78.02±2.22	79.19±0.76
	✗	✓	69.82±2.78	78.83±2.23	77.58±2.30
	✓	✗	73.05±1.87	79.56±1.97	79.26±1.27
	✓	✓	**75.61±1.92**	**83.41±4.40**	**80.94±0.92**
STL10	✗	✗	83.23±0.49	91.80±0.27	84.15±0.72
	✗	✓	82.71±1.19	91.50±0.65	84.04±0.33
	✓	✗	84.70±0.99	92.57±0.53	85.09±0.85
	✓	✓	**85.92±0.04**	**93.23±0.02**	**86.07±0.02**
ImageNet-10	✗	✗	75.79±2.19	81.88±2.08	82.51±1.37
	✗	✓	75.54±2.17	80.86±1.97	**82.73±1.39**
	✓	✗	74.69±1.24	79.75±1.54	82.09±0.77
	✓	✓	**77.25±3.75**	**86.21±4.03**	81.68±1.99
HHAR	✗	✗	73.84±1.85	85.12±1.14	80.70±1.01
	✗	✓	75.90±1.55	87.06±1.30	81.13±1.01
	✓	✗	77.12±0.84	88.07±0.62	82.40±0.68
	✓	✓	**78.20±0.33**	**88.95±0.21**	**82.42±0.36**
REUT	✗	✗	53.37±8.33	77.22±4.58	50.96±6.06
	✗	✓	61.26±1.34	80.71±0.69	57.92±1.62
	✓	✗	63.08±0.48	81.84±0.48	59.11±0.19
	✓	✓	**63.53±0.66**	**81.90±0.16**	**60.32±0.40**
ACM	✗	✗	63.87±5.98	86.07±2.81	59.64±4.46
	✗	✓	73.58±1.55	90.31±0.62	68.42±1.30
	✓	✗	72.60±2.51	89.79±1.10	68.29±2.07
	✓	✓	**76.04±0.39**	**91.30±0.17**	**70.40±0.30**
CITE	✗	✗	43.10±2.49	69.06±1.65	42.79±1.30
	✗	✓	47.21±0.50	71.36±0.32	44.09±0.49
	✓	✗	44.68±1.23	70.15±0.71	43.60±0.77
	✓	✓	**48.32±0.57**	**72.27±0.37**	**45.77±0.48**
DBLP	✗	✗	51.66±3.06	77.62±2.05	47.04±1.87
	✗	✓	54.64±0.59	79.79±0.38	49.57±0.33
	✓	✗	48.20±1.14	75.81±0.73	43.67±0.82
	✓	✓	**56.39±0.93**	**80.53±0.54**	**50.85±0.60**
PubMed	✗	✗	28.47±1.67	67.95±0.99	28.31±1.64
	✗	✓	30.39±1.15	68.84±0.94	30.08±1.25
	✓	✗	27.20±1.69	66.85±0.78	27.87±1.84
	✓	✓	**34.45±0.58**	**71.91±0.22**	**32.15±1.81**

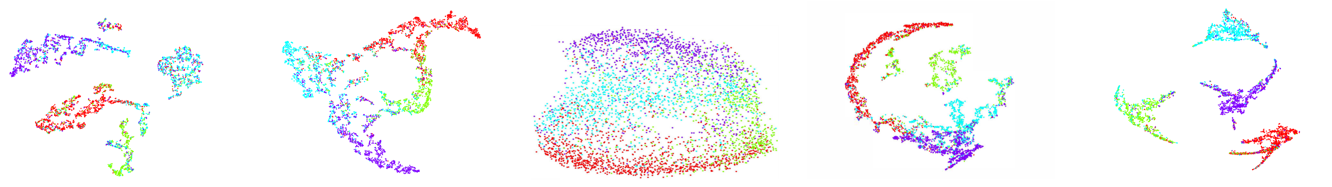
I. Visual Comparison

To qualitatively validate the significant performance of the proposed method EGRC-Net, we plotted t-SNE visualizations [60] of the learned embeddings by SDCN [19], AGCN [20], EGAE [23], AGCC [21], and our EGRC-Net on DBLP. The visualized results are shown in Figure 8, where we can observe that the representation resulting from our method



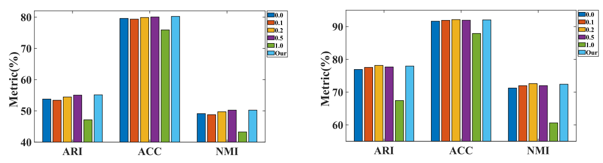
(a) epoch = 1 (b) epoch = 50 (c) epoch = 100 (d) epoch = 150 (e) epoch = 200

Fig. 7. Visualization of the learned representations w.r.t. the epoch number on REUT, where different colors represent different clusters, the homogeneity and completeness measure the visualized quality of the embeddings with a higher score indicating a better representation.



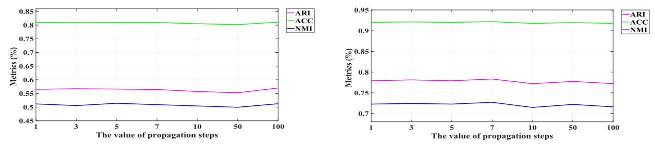
(a) SDCN (b) AGCN (c) EGAE (d) AGCC (e) Our

Fig. 8. Visualization of the learned representations by (a) SDCN [19], (b) AGCN [20], (c) EGAE [23], (d) AGCC [21], and (e) our EGRC-Net on DBLP, where different colors represent different clusters.



(a) DBLP (b) ACM

Fig. 9. The clustering results on (a) DBLP and (b) ACM corresponding to the fixed teleport probability values (i.e., $\rho = 0.0, 0.1, 0.2, 0.5, 1.0$ in Eq. (3)) and our learned one (i.e., Θ in Eq. (16)), where our learned one always obtains the best clustering performance among all metrics.



(a) DBLP (b) ACM

Fig. 10. The clustering results on (a) DBLP and (b) ACM with three metrics w.r.t. the number of propagation steps.

shows the best separability within different clusters. That is to say, the intra-cluster samples gather together, and gaps among inter-clusters are obvious, illustrating that our proposed method EGRC-Net can provide a better discriminative representation than state-of-the-art methods.

J. Analysis of Scalable EGRC-Net

We investigated the effect of the teleport probability value corresponding to the typical fixed value (i.e., ρ in Eq. (3)) and our learned one (i.e., Θ in Eq. (16)). Figure 9 gets the clustering results on DBLP and ACM with a series of threshold settings (i.e., $\rho = 0.0, 0.1, 0.2, 0.5, 1.0$ and our learned one), where different fixed heuristic threshold values show different clustering performances while our learned teleport probability value always obtains the best clustering performance among all metrics. In addition, we can find that the case without the topology structure information (i.e., $\rho = 1.0$) makes a great performance degradation, illustrating the importance of considering graph information in the clustering task.

Additionally, we investigated different numbers of propagation steps τ on DBLP and ACM in Figure 10, where the scalable EGRC-Net performs stable clustering results in three metrics, which has the same effect as APPNP [28] of utilizing far more propagation steps without leading to over smoothing. As also shown in that figure, it is enough to use one power iteration to effectively complete the transmission and aggregation of node information on the graph.

TABLE V

THE COMPARISONS OF STATE-OF-THE-ART APPROACHES AND THE PROPOSED METHODS ON TRAINING TIME AND NETWORK PARAMETERS AS WELL AS THE CLUSTERING PERFORMANCE, WHERE THE ↑ SHOWS THE CLUSTERING IMPROVEMENTS OVER THE BEST COMPARISONS WITH THREE METRICS. WE HIGHLIGHTED THE BEST RESULTS WITH **BOLD**

Dataset		ARI(%)	ACC(%)	NMI(%)	Time(s)	Parameter(M)
ACM	SDCN	73.91±0.40	90.45±0.18	68.31±0.25	**265.930**	6.622940
	AGCN	74.20±0.38	90.59±0.15	68.38±0.45	502.224	6.659081
	AGCC	73.73±0.90	90.38±0.38	68.34±0.89	2204.842	14.934554
	Our	76.04±0.39	91.30±0.17	70.40±0.30	1307.355	6.671183
	Our (Scalable)	**77.94±0.49**	**92.04±0.19**	**72.41±0.47**	1224.115	**4.435626**
	boost ↑	↑ 3.74	↑ 1.45	↑ 4.03		
CITE	SDCN	40.17±0.43	65.96±0.31	38.71±0.32	**296.461**	9.374333
	AGCN	43.79±0.31	68.79±0.23	41.54±0.30	512.913	9.419504
	AGCC	41.82±2.03	68.08±1.44	40.86±1.45	3230.083	18.608447
	Our	**48.32±0.57**	**72.27±0.37**	**45.77±0.48**	1726.946	9.432814
	Our (Scalable)	34.38±0.93	62.71±0.74	35.15±0.65	1524.338	**6.288308**
	boost ↑	↑4.53	↑ 3.48	↑ 4.23		
DBLP	SDCN	39.15±2.01	68.05±1.81	39.50±1.34	**299.241**	4.317424
	AGCN	42.49±0.31	73.26±0.37	39.68±0.42	648.020	4.356575
	AGCC	44.40±3.79	73.45±2.16	40.36±2.81	2539.032	11.863038
	Our	**56.39±0.93**	**80.53±0.54**	**50.85±0.60**	2430.525	4.372805
	Our (Scalable)	55.13±1.67	80.23±0.86	50.23±1.12	1951.387	**2.897955**
	boost ↑	↑ 11.99	↑ 7.08	↑ 10.49		

K. Running Time and Parameters Analysis

We trained SDCN [19], AGCN [20], AGCC [21], our EGRC-Net, and the scalable variant for 200 epochs on ACM, CITE, and DBLP. The experiments were repeated ten times. The compared results regarding the training time are shown in Table V, where the comparisons show that our EGRC-Net achieves a noteworthy improvement in clustering performance while maintaining acceptable resource consumption. Furthermore, in the case of DBLP, the scalable variant demonstrates a 10.73% increase in ARI while reducing running time by 19.71% and memory usage by 33.73%. Particularly, the performance of the scalable one in CITE is not significant. The reason is possible that in the graph of CITE, many nodes are not well-connected, making a limited graph embedding learning capability with the IAPPNP module. Notably, how to ensure the scalability of a graph neural network while ensuring its effectiveness is still an open and challenging research topic.

V. Conclusion

We presented a novel graph refinement clustering network to address the problem that existing GCN-based graph clustering networks heavily rely on a predefined graph and may fail if the initial graph cannot truly and precisely reflect their topology structures on the embedding space. Specifically, we leveraged the vanilla DAE and GCN modules, a series of MLPs, and a graph fusion module to conduct embedding-induced graph refinement. Afterward, we minimized the Jeffreys divergence between multiple derived distributions to jointly optimize the embedding learning, the graph refinement, and the clustering assignments to achieve the clustering performance. In addition, we also designed a simple yet effective graph embedding learning module (i.e., improved APPNP) to replace the vanilla GCN, resulting in a scalable variant of EGRC-Net. Extensive experiments and analyses on nine benchmark datasets with fifteen compared methods demonstrated that our EGRC-Net could consistently outperform state-of-the-art approaches. A series of experiments and analyses were conducted to understand the effectiveness of the proposed methods. In future work, we will focus on the interoperability of this method, contributing to the graph neural network community.

References

[1] K. Zhan, F. Nie, J. Wang, and Y. Yang, "Multiview consensus graph clustering," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1261–1270, Mar. 2019.

[2] P. Li, H. Zhao, and H. Liu, "Deep fair clustering for visual learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9067–9076.

[3] Z. Peng, H. Liu, Y. Jia, and J. Hou, "Adaptive attribute and structure subspace clustering network," *IEEE Trans. Image Process.*, vol. 31, pp. 3430–3439, 2022.

[4] Y. Jia, G. Lu, H. Liu, and J. Hou, "Semi-supervised subspace clustering via tensor low-rank representation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 7, pp. 3455–3461, Jul. 2023.

[5] W. Xia, T. Wang, Q. Gao, M. Yang, and X. Gao, "Graph embedding contrastive multi-modal representation learning for clustering," *IEEE Trans. Image Process.*, vol. 32, pp. 1170–1183, 2023.

[6] P. Ghosh, N. Saini, L. S. Davis, and A. Shrivastava, "Learning graphs for knowledge transfer with limited labels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 11146–11156.

[7] X. Wang, S. Fan, K. Kuang, C. Shi, J. Liu, and B. Wang, "Decorrelated clustering with data selection bias," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 2177–2183.

[8] L. Gong, S. Zhou, W. Tu, and X. Liu, "Attributed graph clustering with dual redundancy reduction," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, Jul. 2022, pp. 1–7.

[9] X. Zhang, H. Liu, X. Zhang, and X. Liu, "Attributed graph clustering with multi-task embedding learning," *Neural Netw.*, vol. 152, pp. 224–233, Aug. 2022.

[10] L. Miklautz, D. Mautz, M. C. Altinigneli, C. Bohm, and C. Plant, "Deep embedded non-redundant clustering," in *Proc. AAAI*, vol. 34, 2020, pp. 5174–5181.

[11] Z. Peng, Y. Jia, H. Liu, J. Hou, and Q. Zhang, "Maximum entropy subspace clustering network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 2199–2210, Apr. 2022.

[12] Y. Jia, H. Liu, J. Hou, and Q. Zhang, "Clustering ensemble meets low-rank tensor approximation," in *Proc. AAAI*, 2021, vol. 35, no. 9, pp. 7970–7978.

[13] L. Yu et al., "SAIL: Self-augmented graph contrastive learning," in *Proc. AAAI*, 2022, pp. 8927–8935.

[14] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.

[15] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. ICML*, New York, NY, USA, 2016, pp. 478–487.

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

[17] T. N. Kipf and M. Welling, "Variational graph auto-encoders," in *Proc. NIPS Workshop*, 2016, pp. 1–3.

[18] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang, and C. Zhang, "Attributed graph clustering: A deep attentional embedding approach," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Macao, China, Aug. 2019, pp. 3670–3676.

[19] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, and P. Cui, "Structural deep clustering network," in *Proc. Web Conf.*, New York, NY, USA. Taipei, Taiwan: Association for Computing Machinery, Apr. 2020, pp. 1400–1410.

[20] Z. Peng, H. Liu, Y. Jia, and J. Hou, "Attention-driven graph clustering network," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 935–943.

[21] X. He, B. Wang, Y. Hu, J. Gao, Y. Sun, and B. Yin, "Parallelly adaptive graph convolutional clustering model," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, May 26, 2022, doi: 10.1109/TNNLS.2022.3176411.

[22] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, "Adversarially regularized graph autoencoder for graph embedding," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 1–7.

[23] H. Zhang, P. Li, R. Zhang, and X. Li, "Embedding graph auto-encoder for graph clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 11, pp. 9352—9362, Nov. 2023.

[24] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, vol. 1, 1967, pp. 281–297.

[25] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 974–983.

[26] N. Park et al., "CGC: Contrastive graph clustering forCommunity detection and tracking," in *Proc. ACM Web Conf.*, Apr. 2022, pp. 1115–1126.

[27] A. Bojchevski et al., "Scaling graph neural networks with approximate pagerank," in *Proc. ACM SIGKDD*, 2020, pp. 2464–2473.

[28] J. Gasteiger, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," in *Proc. ICLR*, 2019, pp. 1–15.

[29] S. Naumov, G. Yaroslavtsev, and D. Avdiukhin, "Objective-based hierarchical clustering of deep embedding vectors," in *Proc. AAAI*, 2021, pp. 9055–9063.

[30] A. Lin, A. H. Song, and D. Ba, "Mixture model auto-encoders: Deep clustering through dictionary learning," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2022, pp. 3368–3372.

[31] C. Niu, H. Shan, and G. Wang, "SPICE: Semantic pseudo-labeling for image clustering," *IEEE Trans. Image Process.*, vol. 31, pp. 7264–7278, 2022.

[32] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML*, vol. 30. Atlanta, GA, USA, 2013, p. 3.

[33] E. Pan and Z. Kang, "Multi-view contrastive graph clustering," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 2148–2159.

[34] M. Xu, P. Fu, B. Liu, and J. Li, "Multi-stream attention-aware graph convolution network for video salient object detection," *IEEE Trans. Image Process.*, vol. 30, pp. 4183–4197, 2021.

[35] H. Zhang, J. Shi, R. Zhang, and X. Li, "Non-graph data clustering via O(n) bipartite graph convolution," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 7, pp. 8729–8742, Jul. 2023.

[36] T. Chen et al., "Bag of tricks for training deeper graph neural networks: A comprehensive benchmark study," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 2769–2781, Mar. 2023.

[37] R. Zhang, W. Zhang, P. Li, and X. Li, "Graph convolution RPCA with adaptive graph," *IEEE Trans. Image Process.*, vol. 31, pp. 6062–6071, 2022.

[38] W. Wang, G. Zhang, H. Han, and C. Zhang, "Correntropy-induced Wasserstein GCN: Learning graph embedding via domain adaptation," *IEEE Trans. Image Process.*, vol. 32, pp. 3980–3993, 2023.

[39] H. Zhang, Y. Zhu, and X. Li, "Towards projected clustering with aggregated mapping," *IEEE Trans. Image Process.*, vol. 32, pp. 4103–4113, 2023.

[40] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. ICLR*, Vancouver, BC, Canada, 2018, pp. 1–12.

[41] G. Salha, R. Hennequin, J.-B. Remy, M. Moussallam, and M. Vazirgiannis, "FastGAE: Scalable graph autoencoders with stochastic subgraph decoding," *Neural Netw.*, vol. 142, pp. 1–19, Oct. 2021.

[42] N. Mrabah, M. Bouguessa, M. F. Touati, and R. Ksantini, "Rethinking graph auto-encoder models for attributed graph clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 9, pp. 9037–9053, Sep. 2023.

[43] G. Salha-Galvan, J. F. Lutzeyer, G. Dasoulas, R. Hennequin, and M. Vazirgiannis, "Modularity-aware graph autoencoders for joint community detection and link prediction," *Neural Netw.*, vol. 153, pp. 474–495, Sep. 2022.

[44] L. Wu, P. Cui, J. Pei, and L. Zhao, *Graph Neural Networks: Foundations, Frontiers, and Applications*. Singapore: Springer, 2022.

[45] Z. Peng, H. Liu, Y. Jia, and J. Hou, "Deep attention-guided graph clustering with dual self-supervision," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 7, pp. 3296–3307, Jul. 2023.

[46] I. Spinelli, S. Scardapane, and A. Uncini, "Adaptive propagation graph convolutional network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 10, pp. 4755–4760, Oct. 2021.

[47] Z. Hou, Y. Cen, Y. Dong, J. Zhang, and J. Tang, "Automated unsupervised graph representation learning," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 3, pp. 2285–2298, Mar. 2023.

[48] J. Choi, "Personalized pagerank graph attention networks," in *Proc. ICASSP*, 2022, pp. 3578–3582.

[49] J. Topping, F. D. Giovanni, B. P. Chamberlain, X. Dong, and M. M. Bronstein, "Understanding over-squashing and bottlenecks on graphs via curvature," in *Proc. ICLR*, 2022, pp. 1–30.

[50] F. Helmert, "Die genauigkeit der formel von peters zur berechnung des wahrscheinlichen beobachtungsfehlers director beobachtungen gleicher genauigkeit," *Astronomische Nachrichten*, vol. 88, nos. 8–9, pp. 113–131, Jan. 1876.

[51] Student, "The probable error of a mean," *Biometrika*, vol. 6, no. 1, pp. 1–25, Mar. 1908.

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

[53] Y. Le Cun et al., "Handwritten zip code recognition with multilayer networks," in *Proc. ICPR*, vol. 2, 1990, pp. 35–40.

[54] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. ICAIS*, 2011, pp. 215–223.

[55] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[56] A. Stisen et al., "Smart devices are different: Assessing and Mitigating-Mobile sensing heterogeneities for activity recognition," in *Proc. 13th ACM Conf. Embedded Networked Sensor Syst.*, New York, NY, USA, Nov. 2015, pp. 127–140.

[57] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "RCV1: A new benchmark collection for text categorization research," *J. Mach. Learn. Res.*, vol. 5, pp. 361–397, Dec. 2004.

[58] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–106, Sep. 2008.

[59] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[60] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[61] W. Wang, C. Yang, H. Chen, and X. Feng, "Unified discriminative and coherent semi-supervised subspace clustering," *IEEE Trans. Image Process.*, vol. 27, no. 5, pp. 2461–2470, May 2018.

Zhihao Peng received the B.S. and M.S. degrees in computer science and technology from the Guangdong University of Technology, Guangzhou, China, in 2016 and 2019, respectively, and the Ph.D. degree in computer science from the City University of Hong Kong, SAR, China, in 2023.

He is currently a Postdoctoral Fellow of electrical engineering with The Chinese University of Hong Kong. His current research interests include spectral clustering, subspace learning, and domain adaptation in image processing with unsupervised learning.

Hui Liu received the B.Sc. degree in communication engineering from Central South University, Changsha, China, the M.Eng. degree in computer science from Nanyang Technological University, Singapore, and the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong. From 2014 to 2017, she was a Research Associate with the Maritime Institute, Nanyang Technological University. She is currently an Assistant Professor with the School of Computing Information Sciences, Caritas Institute of Higher Education, Hong Kong. Her current research interests include image processing and machine learning.

Yuheng Jia (Member, IEEE) received the B.S. degree in automation and the M.S. degree in control theory and engineering from Zhengzhou University, Zhengzhou, China, in 2012 and 2015, respectively, and the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, China, in 2019.

He is currently an Associate Professor with the School of Computer Science and Engineering, Southeast University, Nanjing, China. His current research interests include machine learning and data representation, such as weakly-supervised learning, high-dimensional data modeling and analysis, and low-rank tensor/matrix approximation and factorization.

Junhui Hou (Senior Member, IEEE) received the B.Eng. degree in information engineering (talented students program) from the South China University of Technology, Guangzhou, China, in 2009, the M.Eng. degree in signal and information processing from Northwestern Polytechnical University, Xi'an, China, in 2012, and the Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, in 2016.

He is currently an Associate Professor with the Department of Computer Science, City University of Hong Kong. His current research interests include multi-dimensional visual computing. He is an Elected Member of IEEE MSA-TC, VSPC-TC, and MMSP-TC. He received the Early Career Award (3/381) from the Hong Kong Research Grants Council in 2018. He is currently serving as an Associate Editor for IEEE Transactions on Visualization and Computer Graphics, IEEE Transactions on Circuits and Systems for Video Technology, IEEE Transactions on Image Processing, *Signal Processing: Image Communication*, and *The Visual Computer*.