2024 International Conference on Networking and Network Applications (NaNA) | 979-8-3503-7677-7/24/$31.00 ©2024 IEEE | DOI: 10.1109/NaNA63151.2024.00070

Deep Attention Fusion Network for Attributed Graph Clustering

Congrui Wang, Li Li*, Fei Hao
School of Computer Science, Shaanxi Normal University, Xi'an 710119, China
E-mails: {wangcr, lili, fhao}@snnu.edu.cn

***Abstract*—The field of graph clustering has gained prominence in data mining and machine learning due to the surge in big data and social networks. Deep clustering networks, which are commonly used for graph clustering, extract information from both attributes and structures to perform clustering tasks. However, the heterogeneity between the structure and attribute features of nodes can limit the model's clustering performance. To address this, a deep attention fusion network is proposed for attribute graph clustering. The network comprises an autoencoder and a graph autoencoder, which independently extract the attribute and structural features from the attribute graph. An attention-based information fusion module is also introduced. This module links the autoencoder and the graph autoencoder layer by layer, enhancing the fusion of the two types of heterogeneous information between layers. Additionally, the information enhancement module, which has the capability to further enhance the final embedding vector, contributes to the overall effectiveness of the model. The model is trained using self-supervised learning on the attribute graph dataset. The results of the experiment indicated that this method can significantly enhance the clustering performance.**

Keywords-attributed graph clustering; graph convolutional network; attention mechanism; self-supervised learning;

I. INTRODUCTION

The rapid advancement of mobile internet technology in recent years has led to the generation of a vast amount of graph data on the internet. This data has a broad range of applications in real-world scenarios. For example, it can represent mutual following relationships on social networks, the structural data of organic substances or protein structures in biomedicine, and the citation relationships between authors and various papers in citation networks [1-4]. Many of these data sets can be represented as attribute graphs, which include not only the attribute properties of each node but also the structural relationships that denote interactions between nodes. Clustering algorithms, a crucial technology for big data analysis, serve as the starting point for data analysis, providing vital support for subsequent data annotation and organization tasks. The primary task of clustering algorithms is to classify unmarked information through unsupervised means by identifying feature or structural similarities. This ensures that the similarity between objects in the same classification is higher than the similarity between objects in different classifications. Attribute graph clustering, in particular, segregates the nodes in the network into disconnected clusters in an unsupervised manner. It has practical applications in areas such as social network analysis, recommendation systems, and bioinformatics.

The key to the attribute graph clustering task is to obtain the embedded representation of each node that contains clustering information. A number of graph neural networks, including Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT), have been proposed. These methods, based on GNN, have shown impressive performance due to their robust graph structure representation capabilities. Kipf and Welling [5] introduced Graph Auto-Encoder (GAE) and Variational Graph Auto-Encoder (VGAE), applying autoencoders and variational graph autoencoders to graph clustering. VGAE, an unsupervised learning method, employs latent variables to learn the embedded representation of undirected graphs. It uses the known graph encoded by the graph convolutional neural network to learn the distribution of node representations, and then uses the learned embedding to perform the clustering task. Graph data comprises structural information and node feature information. The integration of this information in a unified manner poses a significant challenge. To address this, Wang et al. [6] proposed the Marginalized Graph Auto-Encoder (MGAE), which dynamically adjusts both the topology and content information by introducing random noise.

Attribute graphs carry both structural and attribute information. To capture these two types of information concurrently, and to leverage the benefits of autoencoders [7] and graph neural networks [8], Bo et al. [9] introduced a clustering technique named Structural Deep Clustering Network (SDCN). This method incorporates structural information into deep clustering. SDCN combines the strengths of the AutoEncoder (AE) and Graph Convolutional Network (GCN) through the use of delivery operator and dual self-supervised mechanisms. The delivery operator transmits the embedded representation learned by the autoencoder to the corresponding GCN module. This design allows for the combination of structural and attribute information in the attribute graph, enhancing the graph clustering task. Zhou et al. [10] introduced Deep Fusion Clustering Network. This network integrates the Structure and Attribute Information Fusion module (SAIF), which is based on mutual dependency learning, to achieve more comprehensive and accurate structural representations. While these models have shown promising results, there remains a challenge in the unsupervised graph clustering task. Specifically, there is a heterogeneity difference between the information types of the autoencoder and the graph autoencoder. The effective integration of these types is an issue that still needs attention.

This paper introduces a novel attribute graph clustering network, the Deep Attention Fusion Network. This network employs a graph autoencoder module to capture high-order structural information from the attribute graph, and an autoencoder module to encode attribute information. These

979-8-3503-7677-7/24/$31.00 ©2024 IEEE
DOI 10.1109/NaNA63151.2024.00070

two networks interact and learn from each other to enhance clustering performance. To better integrate these two modules, an attention fusion module is designed. This module fuses the inter-layer information of the two encoders, primarily using an improved attention mechanism. Furthermore, to fully leverage the attribute and structural information of the graph, an information enhancement mechanism is designed to enrich the final vector embedding. Lastly, a self-supervised clustering module is introduced, allowing the model to learn and train based on node representation and cluster allocation.

The main contributions of this paper are as follows:

(1) The proposal of a deep clustering network with attention fusion. This network can simultaneously use the structure and attribute information in the attribute graph for clustering, and learn from each other to enhance the clustering effect.
(2) To better integrate the heterogeneous information of attributes and structures in clustering, we propose an attention fusion module. This module can effectively fuse the interlayer information of autoencoders and graph autoencoders.
(3) The conduction of comparative experiments on four attribute graph datasets, demonstrating the effectiveness of our method.

II. RELATED WORK

A. Graph neural network

Attribute graph clustering is a subset of graph clustering. In an attribute graph, each node is linked to an attribute vector that characterizes the node. The aim of attribute graph clustering is to segregate the nodes in the network into separate clusters in an unsupervised fashion. Graph Neural Networks (GNNs), due to their potent representation capability in graph data structures, have found extensive application in graph clustering. Graph is a complex data structure where any node can form edges with other nodes. The attributes of graph nodes are diverse and describe various features. To handle complex graph structures, graph neural networks primarily come in two types: spectral domain and spatial domain. Kipf et al. [8] were the first to apply the convolution operation in graph data structures. This operation approximates the localization of graph convolution in graph spectral theory to the first order, thereby enabling convolution operations on the graph. Levie et al. [11] later improved the spectral domain convolution using Cayley polynomials. The filters of CayleyNets can identify important narrow frequencies during training, and the resulting spectral filters are spatially localized. They are effective in extracting crucial information from the graph. Spatial domain-based graph neural networks primarily include GraphSage, proposed by Hamilton et al. [12]. The central idea of GraphSage is to learn a function that aggregates representations of neighboring vertices to generate the target vertex's embedding. It is typically used on large-scale graphs. Velickovic et al. [13] proposed a graph attention network that achieves weighted aggregation of domain information by learning the weights of different neighbors. This network exhibits robustness and can address the over-smoothing issue in convolutional neural networks.

B. Attention mechanism

The attention mechanism, first proposed in visual science, refers to the human visual system's tendency to focus on specific parts of a scene to extract key information. This concept was later applied to deep learning, initially in the field of machine translation. Bahdanau et al. [14] utilized an alignment attention mechanism, which allows the translator to connect with the current context when generating each output word. It automatically searches and selects the most relevant parts of the source language sentence and generates output words based on these parts. As the attention mechanism evolved, its working principle can be broadly divided into three steps. First, it calculates the attention score for the input and output, representing the importance of the current output. Next, it uses the SoftMax function to convert the score into weights, ensuring all attention weights lie between 0 and 1 and that their sum is equal to 1. Finally, it multiplies the attention weight by the corresponding vector to obtain the final result. In recent years, the attention mechanism has seen rapid development. The Transformer model [15], which uses an attention mechanism to process sequence data, has achieved impressive results. The Transformer model designs a self-attention mechanism, allowing the model to consider all other elements in the sequence when processing each sequence element. This comprehensive understanding of the context significantly enhances the model's grasp of language structure. Simultaneously, the self-attention mechanism has demonstrated strong performance in information fusion and other domains.

III. OVERALL NETWORK FRAMEWORK

The model we propose, depicted in Figure I, integrates attribute and structural information across layers to enhance clustering performance. It primarily comprises four components: an attribute information encoding module, a graph autoencoder, an attention fusion module, and an information enhancement module. The attribute information encoding module and the graph autoencoder module are used to extract attribute and structural information from the attribute graph, respectively. To more effectively integrate these two types of heterogeneous information, an attention fusion module is employed to link the two encoder layers. The information enhancement module is then used to augment the final embedding, resulting in a more effective clustering representation. This approach ensures a comprehensive and efficient extraction and integration of both attribute and structural information from the graph. X and $\hat{X}$ denote the input attribute features and the reconstruction features, respectively. The "Graph" signifies the graph structure information. $Y^{(\ell)}$ and $Z^{(\ell)}$ are the interlayer representations of the attribute information encoding module and the ℓ-th layer in the graph autoencoder, respectively. $H^{(\ell)}$ is the output information that is fused by the Attention Fusion (AF) module. P and Q represent a self-supervision mechanism that is used to train the model.

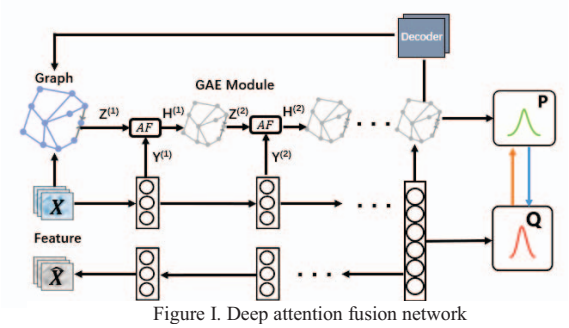

Figure I. Deep attention fusion network

A. Attribute information encoding module

In the attribute graph, each node possesses attribute information that characterizes it. Our model employs a fully connected layer to distill useful information from this complex attribute data. Both the encoder and decoder utilize fully connected layers as the neural network for information extraction. The process is as follows:

$$Y^{(\ell)} = \phi\left(W_e{}^{(\ell)} Y^{(\ell-1)} + b_e{}^{(\ell)}\right) \quad (1)$$

Where ℓ=1,2…L, ϕ is the activation function of the fully connected layer, such as the Relu or Sigmoid function. W and b are the weight matrix and bias of the ℓ layer in the encoder, respectively. In addition, we represent $Y^{(0)}$ as the attribute matrix X of the original data. It takes the attributes of each node in the attribute graph as input and compresses it into a low-dimensional encoding, capturing important attribute information in the input data. The generated embedding vector will be used for subsequent clustering tasks.

$$L_{AE} = \frac{1}{2N}\sum_{i=1}^{N} ||x_i - \hat{x}_i||_2^2 \quad (2)$$

The training objective of the autoencoder is to minimize the difference between the input data and the data restored in the decoder, and this difference is usually represented by the Mean Squared Error (MSE).

B. Graph autoencoder

To fully leverage the structural information between nodes in the attribute graph, we employ Graph Neural Networks for feature extraction. The graph encoder, a symmetric network, learns the graph's feature representation through an encoder-decoder mechanism. The encoder encodes the input graph data and extracts the feature representation of the graph structure information. The decoder then reconstructs the original graph data from the encoded features. This process is typically implemented using Graph Convolutional Networks (GCN). The operation of each graph convolutional layer can be represented as follows:

$$Z^{(\ell)} = \phi\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}Z^{(\ell-1)}W^{(\ell-1)}\right) \quad (3)$$

Here, A represents the adjacency matrix, $\tilde{A} = A + I$, where I is the identity matrix. W represents the weight, $\tilde{D}$ represents the degree matrix, and $Z^{(\ell)}$ represents the output of the ℓ layer of GCN, which is also the input of the next layer. Specifically, the input of the first layer is the attribute matrix X.

After processing through multiple layers of the graph neural network, the encoder transforms the original adjacency matrix A and attribute matrix X into an embedding vector Z for each node. To optimize the model's training, we employ the following formula as the loss function:

$$L_{GAE} = L_f + L_a = \frac{1}{2N}\left\|\tilde{A}X - \hat{Z}\right\|_2^2 + \frac{1}{2N}\left\|\tilde{A} - \hat{A}\right\|_2^2 \quad (4)$$

The two losses correspond to the loss of the reconstructed feature matrix and the reconstructed adjacency matrix, respectively. Both utilize the MSE as the loss function. The training objective aims to minimize the discrepancy between the input feature matrix and the reconstructed feature matrix, as well as the difference between the original adjacency matrix and the reconstructed adjacency matrix.

C. Attention fusion module

As mentioned above, we use an autoencoder to extract attribute features and a graph autoencoder to extract structural features. If only these two parts are trained separately, it will cause the model to be relatively fragmented and not holistic. To address this, we introduce an attention fusion module that connects the two types of encoder layers, making the model interactive and better integrating the two types of information for clustering.

$$H^{(\ell)} = \alpha Z^{(\ell)} + (1-\alpha)Y^{(\ell)} \quad (5)$$

Firstly, we proportionally add the inter-layer values of the two encoders, where $Z^{(\ell)}$ is the inter−layer value of the graph encoder, $Y^{(\ell)}$ is the inter-layer value of the autoencoder, and α is a learnable parameter with an initial value set to 0.5, which can be automatically adjusted in stochastic gradient descent.

$$Q = W^q H \quad (6)$$

$$K = W^k H \quad (7)$$

$$V = W^v H \quad (8)$$

For the obtained vector H, we multiply it by three weight matrices respectively, these three weight matrices are randomly initialized and continuously updated with deep learning training. Finally, we will get the Query, Key, and Value vectors.

$$\alpha_{i,j} = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \quad (9)$$

$$\alpha' = \alpha M \quad (10)$$

Then we calculate the attention weight score. Here K^T is the transpose of K, d_k is the dimension of the Key vector, after calculating the product of Query and Key, we use the SoftMax function to convert it into a score value. In addition, in order to further integrate different position information, we use a weighted matrix M to weight each point of the obtained attention score once. By weighting each point, the attention score can be more precise.

$$b_i = \sum_{j=0}^{n} v_j \alpha'_{i,j} \quad (11)$$

Finally, we multiply each element's Value vector by its corresponding attention weight and sum them to get the final output.

D. Information enhancement module

To fully harness the attribute information and topological information on the attribute graph, we've designed an information enhancement module. This module primarily consists of two parts. First, we linearly combine the embedding vectors finally obtained by the autoencoder and the graph autoencoder. Similar to the previous AF module, here β is also a learnable parameter, and we set the initial value to 0.5.

$$Z_I = \beta Z_{AE} + (1-\beta) Z_{GAE} \tag{12}$$

$$Z_L = \tilde{A} Z_I \tag{13}$$

Then we use operations akin to graph convolution to enhance the information, and the final Z_L is the vector used for clustering. Through these operations, the final embedding vector is effectively enhanced, enabling the model to learn more comprehensive and useful information for clustering.

E. Self-supervised training

During the model training process, there's no label guidance. To accomplish the graph clustering training task in an unsupervised manner, we introduce self-supervised training. First, we obtain the clustering distribution Q, and then we derive the distribution P from Q. The ultimate goal is to make Q and P closer through the loss function.

To initialize the clustering center, we first employ the k-means algorithm to pre-cluster the original data, which yields a soft label for clustering. We represent this soft distribution label using the t-distribution:

$$q_{ij} = \frac{\left(1 + \frac{\left\|z_i - c_j\right\|^2}{t}\right)^{-\frac{t+1}{2}}}{\sum_k \left(1 + \frac{\|z_i - c_k\|^2}{t}\right)^{-\frac{t+1}{2}}} \tag{14}$$

Here q_{ij} represents the soft assignment of node i to class c_j, which essentially represents the probability. The assignments generated in this distribution are deemed reliable, so we generate distribution P through square normalization, and its calculation formula is as follows.

$$p_{ij} = \frac{\frac{q_{ij}^2}{f_j}}{\sum_m \left(\frac{q_{im}^2}{f_m}\right)} \tag{15}$$

In every cycle of deep learning training, we will get two distributions, P and Q. Since P is more reliable than Q, we employ KL divergence to optimize the two distributions, bringing them closer together. This process ensures that the learned node embedding contains more information beneficial to clustering.

$$L_{cl} = KL(P||Q) = \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{16}$$

IV. EXPERIMENTAL

In this section, we will conduct experiments and analyses on four commonly used real-world graph datasets to further verify the effectiveness of the model.

A. Datasets

ACM: ACM represents a paper network dataset. It consists of 3025 papers from various categories in the ACM database. An edge exists between two papers in the graph data if they share the same author. This dataset encompasses three different categories.

DBLP: DBLP is a dataset comprising 4058 paper authors. An edge is formed if two authors have collaborated academically. It includes four research fields: machine learning, information retrieval, data mining, and databases.

Citeseer: This dataset contains 3327 scientific publications and 4732 edges. Each publication in the dataset is represented by a binary vector, indicating the presence or absence of the corresponding word in the article.

Cora: This dataset is composed of papers in the field of machine learning. Each paper cites or is cited by at least one other paper. It contains a total of 2708 papers, divided into seven different categories.

TABLE I The statistics of the datasets

Dataset	Node	Edge	Dimension	Classes
ACM	3025	13128	1870	3
DBLP	4058	3528	334	4
Citeseer	3327	4732	3703	6
Cora	2708	5429	1433	7

B. Compared methods

Kmeans: A classic clustering algorithm in the field of deep learning.

Auto-Encoder (AE): This method uses an autoencoder to extract data features, and then completes clustering through traditional clustering algorithms.

GAE[8]: Employs a graph neural network as an encoder to learn node feature representations and clusters in an unsupervised manner.

AGC[16]: Utilizes k-order graph convolution to capture high-order structural information in attribute graphs, and can adaptively select the order.

DAEGC[17]: Uses a deep graph attention network to obtain node embeddings, and then optimizes the clustering results using a loss function with a target distribution.

SDCN[9]: Designs a dual-channel network that can separately obtain structural and attribute information from attribute graphs, and optimizes clustering allocation and representation learning in a unified framework.

DFCN[10]: Fuses the representations learned by AE and GAE through the attribute information fusion module, and completes training through multiple supervisions.

C. Experimental parameters and indicators

To evaluate all methods comprehensively, we've selected four widely used clustering evaluation metrics: Accuracy (ACC), Normalized Mutual Information (NMI), Average Rand Index (ARI), and Macro F1-score (F1). For all these metrics, a higher score indicates better clustering performance.

For the four datasets—ACM, Citeseer, DBLP, and Cora—the learning rates for the experiments are set to *9e-5, 8e-5, 3e-3*, and *2e-4*, respectively. The number of training epochs is set to *200* for all. The experiments are conducted on an NVIDIA GTX 3090 graphics card with 32GB of memory, under a Linux operating system.

TABLE II Clustering Results on Four Datasets

Dataset	Metric	Kmeans	AE	GAE	AGC	DAEGC	SDCN	DFCN	OUR
ACM	ACC	67.31	81.83	84.52	83.12	86.94	90.45	90.90	92.43
	NMI	32.44	49.30	55.38	53.98	56.18	68.31	69.40	73.29
	ARI	30.60	54.64	59.46	56.20	59.35	73.91	74.90	78.87
	F1	67.57	82.01	84.65	83.33	87.07	90.42	90.80	92.45
DBLP	ACC	38.65	51.43	61.21	55.16	62.05	68.05	76.00	77.99
	NMI	11.45	25.40	30.30	26.92	32.49	39.50	43.70	46.71
	ARI	6.97	12.21	22.02	13.86	21.03	39.15	47.00	51.48
	F1	31.92	52.53	61.41	53.22	61.75	67.71	75.70	77.27
Citeseer	ACC	39.32	57.08	61.35	68.04	64.54	65.96	69.50	70.27
	NMI	16.94	27.64	34.64	41.90	36.41	38.71	43.90	44.73
	ARI	13.43	29.31	33.55	43.18	37.78	40.17	45.00	45.56
	F1	36.08	53.80	57.36	63.48	62.20	63.62	64.30	64.81
Cora	ACC	36.07	44.63	63.80	68.86	67.21	50.70	56.87	72.38
	NMI	17.04	23.59	47.64	53.16	50.63	33.78	38.81	53.90
	ARI	9.41	19.24	38.00	44.37	47.33	25.76	29.79	51.18
	F1	31.14	42.68	65.86	65.59	60.82	44.13	55.92	63.61

D. Experiment results analysis

The experimental results demonstrate that methods using Graph Convolutional Networks (GCN) outperform those using AutoEncoders (AE) and Kmeans. This is because AE and Kmeans methods only utilize the attribute information of the data, neglecting to fully use the structural information. This underscores the crucial role of graph neural networks in attribute graph clustering.

We also observed that the Structural Deep Clustering Network (SDCN) method shows a significant improvement compared to other GCN methods in three datasets. SDCN can incrementally incorporate content information into the structural representation, which can better aid the clustering task. This suggests that both attribute information and structural information are vital in clustering tasks.

Our method outperforms the SDCN method in four datasets and is significantly superior to the Deep Fusion Clustering Network (DFCN) method in the ACM, DBLP, and Cora datasets. For instance, in the ACM dataset, our method has improved by 3.89% and 3.97% respectively in the NMI and ARI indicators compared to DFCN. This is because our method can more effectively integrate attribute and structural information to achieve superior clustering results. The experimental results also highlight the importance of integrating attribute and structural information for attribute graph clustering. The improvement effect is less pronounced in the Citeseer dataset, which may be due to the lengthy dimension of attribute information in the Citeseer dataset affecting the fusion effect. We aim to improve this in future work.

V. CONCLUSION

This thesis introduces an attribute graph clustering network with attention fusion. The model primarily comprises an autoencoder and a graph autoencoder that mutually learn. The attention fusion module integrates the information from both parts, addressing the issue of lack of interaction between the two parts in prior methods. Experiments on four real-world datasets demonstrate that our method can effectively amalgamate information to yield improved clustering results. In future work, we plan to explore how to better integrate attribute and structural information in datasets with larger attribute information dimensions, such as Citeseer.

VI. ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (61303092), and the Shaanxi Province Natural Science Basic Research Foundation (2020JM-290, 2022JM-371).

REFERENCES

[1] S. Cavallari, V . W. Zheng, H. Cai, K. C.-C. Chang, and E. Cambria,"Learning community embedding with community detection and node embedding on graphs," in Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017.

[2] Y . Wang, Y . Zhang, F. Zhang, S. Wang, M. Lin, Y . Zhang, and X. Sun,"Ada-nets: Face clustering via adaptive neighbour discovery in the structure space," arXiv preprint arXiv:2202.03800, 2022.

[3] H. Xue, V . Mallawaarachchi, Y . Zhang, V . Rajan, and Y . Lin, "Rep-bin: Constraint-based graph representation learning for metagenomicbinning," in Proc. of AAAI, 2022.

[4] H. Alashwal, M. El Halaby, J. J. Crouse, A. Abdalla, and A. A.Moustafa, "The application of unsupervised clustering methods to alzheimer's disease," Frontiers in computational neuroscience, 2019.

[5] T. N. Kipf and M. Welling, "V ariational graph auto-encoders," 2016, arXiv:1611.07308.

[6] C. Wang, S. Pan, G. Long, X. Zhu, and J. Jiang, "MGAE: Marginalized graph autoencoder for graph clustering," in Proc. ACM Conf. Inf. Knowl. Manag., Nov. 2017, pp. 889–898.

[7] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[J]. Advances in neural information processing systems, 2012, 25.

[8] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arXiv preprint arXiv:1609.02907, 2016.

[9] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, and P . Cui, "Structural deep clustering network," in Proc. Web Conf., Apr. 2020, pp. 1400–1410.

[10] Tu W, Zhou S, Liu X, et al. Deep fusion clustering network[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2021, 35(11): 9978-9987.

[11] R. Levie, F. Monti, X. Bresson, and M. M. Bronstein, "CayleyNets:Graph convolutional neural networks with complex rational spectral filters," IEEE Trans. Signal Process., vol. 67, no. 1, pp. 97-109,Jan. 2019.

[12] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in Proc. Neural Inf. Process. Syst.,2017, pp. 1025–1035.

[13] Veličković P, Cucurull G, Casanova A, et al. Graph attention networks[J]. arXiv preprint arXiv:1710.10903, 2017.

[14] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[J]. arXiv preprint arXiv:1409.0473, 2014.

[15] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.

[16] Zhang, X., Liu, H., Li, Q., & Wu, X. M. (2019). Attributed graph clustering via adaptive graph convolution. In Proceedings of the international joint conference on artificial intelligence (pp. 4327–4333).

[17] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang, C. Zhang, Attributed graph clustering: a deep attentional embedding approach, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019, pp. 3670–3676.

[18] J. Hao, W. Zhu, Deep graph clustering with enhanced feature representations for community detection, Appl. Intell. (2022) 1–14.

[19] W. Li, S. Wang, X. Guo, E. Zhu, Deep graph clustering with multi-level subspace fusion, Pattern Recognit. 134 (2023) 109077.

[20] C.T. Duong, T.T. Nguyen, T.-D. Hoang, H. Yin, M. Weidlich, Q.V.H. Nguyen, Deep MinCut: Learning node embeddings by detecting communities, Pattern Recognit.134 (2023) 109126.