IEICE TRANSACTIONS

on Fundamentals of Electronics, Communications and Computer Sciences

DOI:10.1587/transfun.2023EAL2116

Publicized:2024/06/19

A PUBLICATION OF THE ENGINEERING SCIENCES SOCIETY

The Institute of Electronics, Information and Communication Engineers

Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

LETTER

Attributed Graph Clustering Network with Adaptive Feature Fusion

Xuecheng SUN[†], *Nonmember and* **Zheming LU**[†*], *Member*

SUMMARY To fully exploit the attribute information in graphs and dynamically fuse the features from different modalities, this letter proposes the Attributed Graph Clustering Network with Adaptive Feature Fusion (AGC-AFF) for graph clustering, where an Attribute Reconstruction Graph Autoencoder (ARGAE) with masking operation learns to reconstruct the node attributes and adjacency matrix simultaneously, and an Adaptive Feature Fusion (AFF) mechanism dynamically fuses the features from different modules based on node attention. Extensive experiments on various benchmark datasets demonstrate the effectiveness of the proposed method.

key words: *graph clustering, community detection, Graph Neural Network, Graph Autoencoder*

1. Introduction

Graph clustering involves grouping nodes in a graph into clusters or communities based on their attributes and connectivity patterns. It is a research area with increasing practical value [1], and has been widely used in various practical applications, such as community detection in recommendation systems [2], social networks [3] and protein-protein interaction (PPI) networks [4]. However, the unsupervised nature of graph clustering presents greater challenges compared to the supervised tasks. Without the guidance of ground-truth labels, it requires an appropriate loss function and carefully designed network modules to capture the intrinsic clustering patterns of the graph.

Recently, there has been a growing popularity in combining Graph Neural Networks (GNNs) with autoencoders for graph clustering, due to GNNs' powerful capability in capturing the interaction between graph nodes. The first work in this category SDCN [5] trained a fully connected autoencoder to reconstruct the node features, and propagated the features learned by the autoencoder to a Graph Convolutional Network (GCN) module using a weighted summation. The autoencoder features and GCN features were then unified by a dual self-supervised optimization objective to produce consistent prediction results. The following work DFCN [6] proposed a dynamic fusion mechanism to tune the weight for fusing features from different modalities, and a triplet self-supervised strategy to exploit the cross-modality information for a robust target distribution. In R-DGAE [7], the authors presented a graph autoencoder architecture that learned to reconstruct the adjacency matrix, a sampling operator that dropped out the noisy clustering assignments, and a correction mechanism that forced the model to learn clustering-oriented features from the reconstruction task.

However, we observe that most existing GNN-based graph clustering approaches only considering reconstructing the adjacency matrix, leaving the task of reconstructing the node attributes with GNNs unexplored. Also, there is a lack of effective fusion mechanism that integrates the autoencoder features and GNN features. To tackle with these problems, in this letter we propose the Attributed Graph Clustering Network with Adaptive Feature Fusion (AGC-AFF), in which we

- design an Attribute Reconstruction Graph Autoencoder (ARGAE) that simultaneously learns to reconstruct the node attributes and adjacency matrix, and a masking operation that masks the input nodes for the decoder of ARGAE to make the encoded features more robust;
- present an Adaptive Feature Fusion (AFF) mechanism that dynamically fuses the features from different modules based on node attention;
- conduct extensive experiments on various benchmark datasets to demonstrate the effectiveness of the proposed method.

2. Related Work

2.1 Deep Clustering

Deep clustering refers to the technique where deep neural networks are used to perform clustering tasks. Due to the lack of guidance of ground-truth labels, many deep clustering methods focus on designing appropriate optimization objectives for the unsupervised learning process. For example, [8] utilized the loss function of K-means to learn clustering-friendly features, DEC [9] designed a KL divergence loss to increase the cohesion of the predicted clusters, and IDEC [10] improved DEC by adding a reconstruction loss to help the autoencoder learn better embeddings. However, these methods ignore the structure information of the data, thus cannot deal with graph clustering tasks.

2.2 Graph Neural Networks

Recently, Graph Neural Network (GNN) models have become popular for their advantage in capturing the intricate relations within graphs. Graph Convolutional Network (GCN)

[†]The author is with the School of Aeronautics and Astronautics, Zhejiang University, Hangzhou, China.

[*]Corresponding author: zheminglu@zju.edu.cn.

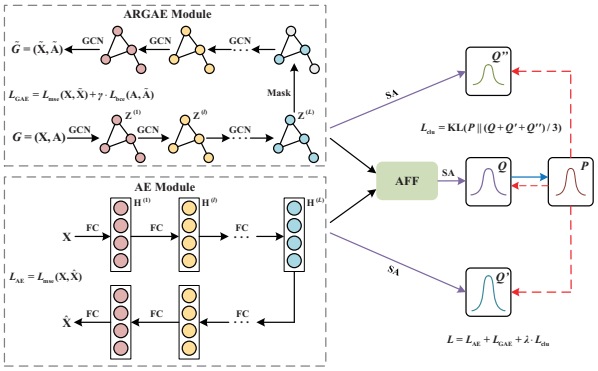

Fig. 1 The pipeline of the proposed AGC-AFF. $\mathbf{X}$ is the input node feature matrix, $\hat{\mathbf{X}}$ is the reconstructed node feature matrix by AE, $\tilde{\mathbf{X}}$ is the reconstructed node feature matrix by ARGAE, $\mathbf{A}$ is the input adjacency matrix, $\tilde{\mathbf{A}}$ is the reconstructed adjacency matrix by ARGAE, $\mathbf{H}$ is the representation encoded by AE, and $\mathbf{Z}$ is the representation encoded by ARGAE. The purple solid line represents soft assignment (SA) generation using Student's t-distribution, the blue solid line represents target distribution generation, and the red dotted line represents the self-supervised training mechanism.

[11] is the most representative GNN model. It leverages the graph structure to perform message passing operations to capture the information across the graph nodes. Graph Attention Network (GAT) [12] extends GCN by incorporating the attention mechanism to enable different weights for different neighbors during node aggregation. Graph Autoencoder (GAE) [13] is designed to learn low-dimensional node representations in a graph by reconstructing the adjacency matrix. In graph clustering, GCN is combined with fully connected autoencoder by SDCN [5] as a regularizer, and GAE is utilized by DGAE [7] for better unsupervised representation learning.

3. Method

In this section, we introduce our Attributed Graph Clustering Network with Adaptive Feature Fusion (AGC-AFF), whose overall framework is shown in Figure 1. AGC-AFF can be divided into four major parts, i.e., autoencoder (AE) module, Attribute Reconstruction Graph Autoencoder (ARGAE) module, Adaptive Feature Fusion (AFF) module and self-supervised training module.

3.1 Autoencoder

The autoencoder (AE) learns representations from the node attributes $\mathbf{X}$. We assume that the encoder of AE consists of L stacked fully connected (FC) layers. Let $\mathbf{H}^{(l)}$ denote the output of the l-th layer, then

$$\mathbf{H}^{(l)} = \phi(\mathbf{W}_a^{(l)}\mathbf{H}^{(l-1)} + \mathbf{b}_a^{(l)}), \tag{1}$$

where ϕ is an activation function, $\mathbf{W}_a^{(l)}$ and $\mathbf{b}_a^{(l)}$ are the weight matrix and bias of the l-th layer, respectively, and $\mathbf{H}^{(0)} = \mathbf{X}$ is the input data. The decoder is the inverse of the encoder, and it takes $\mathbf{H}^{(L)}$ as input and outputs the reconstructed node attributes $\hat{\mathbf{X}}$ also using L stacked FC layers. We can obtain the reconstruction loss of AE by computing an MSE loss:

$$L_{\text{AE}} = L_{\text{mse}}(\mathbf{X}, \hat{\mathbf{X}}) = \frac{1}{N}||\mathbf{X} - \hat{\mathbf{X}}||_{\text{F}}^2, \tag{2}$$

where N is the total number of nodes and $||\cdot||_{\text{F}}$ represents the Frobenius norm.

3.2 Attribute Reconstruction Graph Autoencoder

To fully exploit the attribute information in graphs, we design the Attribute Reconstruction Graph Autoencoder (ARGAE). Let $\mathbf{A}$ denote the adjacency matrix of the graph and $\bar{\mathbf{A}} = \mathbf{A}+\mathbf{I}$, the output of the l-th layer of the encoder $\mathbf{Z}^{(l)}$ can then be obtained by

$$\mathbf{Z}^{(l)} = \phi(\bar{\mathbf{D}}^{-\frac{1}{2}}\bar{\mathbf{A}}\bar{\mathbf{D}}^{-\frac{1}{2}}\mathbf{Z}^{(l-1)}\mathbf{W}_g^{(l)}), \tag{3}$$

where $\mathbf{W}_g^{(l)}$ is the weight matrix of the l-th layer, $\bar{\mathbf{D}}$ is a diagonal matrix with $\bar{\mathbf{D}}_{ii} = \sum_j \bar{\mathbf{A}}_{ij}$, and $\mathbf{Z}^{(0)} = \mathbf{X}$ is the input data. It has been observed that masked autoencoding is beneficial to self-supervised tasks [14], [15], thus we propose a masking operation, which makes the reconstruction target more difficult and helps the model learn more robust features. Specifically, assume $\mathbf{Z}^{(L)}$ is an $N \times d$ matrix, we introduce a learnable vector $\mathbf{a} \in \mathbb{R}^d$ to compute the score of each node:

$$\mathbf{s} = \text{sigmoid}(\mathbf{Z}^{(L)}\mathbf{a}). \tag{4}$$

Then we select k (k is set to $\lceil 0.1N \rceil$ in this work) nodes with the smallest scores and mask their features to $\mathbf{0}$. For the unmasked nodes, we multiply their features with $\mathbf{s}$ to allow the gradients to flow through $\mathbf{a}$:

$$\mathbf{Z}_m = \mathbf{Z}^{(L)} \odot (\mathbf{s}\mathbf{1}^T), \tag{5}$$

where $\odot$ denotes the Hadamard product and $\mathbf{1} \in \mathbb{R}^d$ is an all-ones vector. The decoder is also the inverse of the encoder, and it takes $\mathbf{Z}_m$ as input and outputs the reconstructed node attributes $\tilde{\mathbf{X}}$ and adjacency matrix $\tilde{\mathbf{A}} = \text{sigmoid}(\tilde{\mathbf{X}}\tilde{\mathbf{X}}^T)$. Note that the masking operation may influence the information propagation, thus we adapt the adjacency matrix for the first GCN layer of the decoder to increase the connectivity. Specifically, let idx be the indices of the unmasked nodes, we compute the adapted adjacency matrix by $\mathbf{A}' = \mathbf{A}^2(idx, idx)$. In other words, $\mathbf{A}'$ is the 2nd graph power of $\mathbf{A}(idx, idx)$ that is obtained by performing row and column extractions with idx on the original $\mathbf{A}$. Finally, we can obtain the reconstruction loss of ARGAE by computing an MSE loss and a binary cross-entropy (BCE) loss:

$$L_{\text{GAE}} = L_{\text{mse}}(\mathbf{X}, \tilde{\mathbf{X}}) + \gamma \cdot L_{\text{bce}}(\mathbf{A}, \tilde{\mathbf{A}}), \tag{6}$$

where γ is a strength hyperparameter, and we set it to 0.1 in this work to make the attribute reconstruction loss dominant.

3.3 Adaptive Feature Fusion

To connect the features generated by AE and ARGAE, we

propose the Adaptive Feature Fusion (AFF) mechanism to dynamically fuse the features. Let $\mathbf{w}_f \in \mathbb{R}^{2d}$ be a weight vector, we first concatenate $\mathbf{H}^{(L)}$ and $\mathbf{Z}^{(L)}$ and compute the attention score for each node:

$$\mathbf{m} = \text{softmax}(\phi([\mathbf{H}^{(L)}||\mathbf{Z}^{(L)}]\mathbf{w}_f)). \tag{7}$$

Then we can adaptively fuse the features by

$$\mathbf{F} = \mathbf{H}^{(L)} \odot (\mathbf{m}\mathbf{1}^T) + \mathbf{Z}^{(L)} \odot ((\mathbf{1} - \mathbf{m})\mathbf{1}^T). \tag{8}$$

Note that our AFF can be viewed as an improved version of the FAFGC proposed by [16], where we remove the normalization operation and impose the regularization rule that the sum of the attention scores for $\mathbf{H}^{(L)}$ and $\mathbf{Z}^{(L)}$ equals to $\mathbf{1}$. The advantage of AFF is that it has fewer parameters and is thus less sensitive to overfitting, and in experiments we will show that AFF is the best-performing fusion mechanism in our investigation.

3.4 Self-supervised Training

Following [5], [6], [16], we adopt a Kullback-Leibler (KL) divergence loss between the soft assignments and target distribution to increase the cohesion of the predicted clusters in a self-supervised manner. Let K denote the number of clusters, we first generate the soft assignments $\mathbf{Q} \in \mathbb{R}^{N\times K}$ using the Student's t-distribution:

$$q_{ij} = \text{SA}(\mathbf{f}_i, \boldsymbol{\mu}_j) = \frac{(1 + ||\mathbf{f}_i - \boldsymbol{\mu}_j||^2/v)^{-\frac{v+1}{2}}}{\sum_{j'}(1 + ||\mathbf{f}_i - \boldsymbol{\mu}_{j'}||^2/v)^{-\frac{v+1}{2}}}, \tag{9}$$

where $\mathbf{f}_i$ is the i-th row of $\mathbf{F}$, $\boldsymbol{\mu}_j$ is the j-th cluster center initialized by K-means on representations learned by a pretrained model, and v is the number of degrees of freedom which is set to 1.0 in this work. Similarly, we can compute $q'_{ij} = \text{SA}(\mathbf{h}_i, \boldsymbol{\mu}_j)$ and $q''_{ij} = \text{SA}(\mathbf{z}_i, \boldsymbol{\mu}_j)$ using the features of AE and ARGAE, respectively. The target distribution can be obtained by squaring and normalizing the assignments in $\mathbf{Q}$:

$$p_{ij} = \frac{q_{ij}^2/\sum_i q_{ij}}{\sum_{j'} q_{ij'}^2/\sum_i q_{ij'}}. \tag{10}$$

Since $\mathbf{P}$ has higher confidence, we can minimize the KL divergence between $\mathbf{P}$ and the soft assignments to increase the cohesion of the predicted clusters:

$$L_{\text{clu}} = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{(q_{ij} + q'_{ij} + q''_{ij})/3}, \tag{11}$$

and the overall loss is

$$L = L_{\text{AE}} + L_{\text{GAE}} + L_{\text{clu}}. \tag{12}$$

3.5 Computational Complexity Analysis

Let d_0 denote the dimension of the input data, $d_1, d_2, ...d_L$ denote the output dimensions of the encoding layers of AE and ARGAE, and $|E|$ denote the number of edges of the input graph. The complexity is $O(\sum_{i=1}^{L} Nd_{i-1}d_i)$ for AE, $O(\sum_{i=1}^{L} |E|d_{i-1}d_i + N^2d_0)$ for ARGAE ($O(N^2d_0)$ is imposed by the computation of the predicted adjacency matrix), and $O(NK + N\log N)$ for the computation of the soft assignments. Therefore, the overall computational complexity is $O(\sum_{i=1}^{L} Nd_{i-1}d_i + \sum_{i=1}^{L} |E|d_{i-1}d_i + N^2d_0 + NK + N\log N)$.

4. Experiments

4.1 Datasets

We select five standard graph datasets DBLP, CITE, ACM, CORA and AMAP to evaluate the effectiveness of the proposed method. The detailed statistics can be found in [16].

4.2 Evaluation Metrics

We adopt standard unsupervised evaluation metrics for evaluations and comparisons to other algorithms. Specifically, we use the following four metrics: Accuracy (ACC) [17], Normalized Mutual Information (NMI) [17], Adjusted Rand Index (ARI) [18] and macro F1-score (F1) [19]. ACC can be obtained by calculating the ratio of correctly predicted samples to the total number of samples, and it is a simple and straightforward measure of an algorithm's performance in terms of assigning samples to the correct clusters. NMI is the normalized mutual information between the clustering assignments and the ground truth clusters, and it can provide an understanding of how close the predicted cluster assignments are to the ground truth clustering. ARI is a measure of the similarity between the true and the predicted clusterings, it considers all pairs of samples and counts pairs that are assigned in the same or different clusters in the two clusterings, and adjusts the result by accounting for the chance grouping of samples, which provides a robust evaluation metric for clustering algorithms. F1 is the harmonic mean of the metrics of Precision and Recall, and thus it takes into account the balance between correctly predicting the samples and avoiding misclassification. By using these four metrics together, the effectiveness of the clustering algorithms can be comprehensively demonstrated.

4.3 Implementation Details

The encoders of AE and ARGAE both have four layers with the dimensions of 128, 256, 512 and 20, respectively. AE and ARGAE are pretrained independently for 30 iterations first, and then integrated to a single model and pretrained for another 100 iterations. The integrated model is used to initialize the cluster centers in Eq. (9) and then trained for 200 iterations with the loss in Eq. (12). The learning rate is set to 1e-3 for AMAP, 1e-4 for DBLP, CITE and CORA, and 5e-5 for ACM. The proposed model is implemented by the PyTorch framework and trained on a single NVIDIA RTX 3090 GPU.

Table 1 Comparison with state-of-the-art methods on five datasets. The best results are highlighted with bold and the second-bests are underlined.

Dataset	Metric	SDCN	AGCN	DFCN	GC-SEE	AGC-AFF
DBLP	ACC	68.05	73.26	76.00	79.23	**80.31**
	NMI	39.50	39.68	43.70	48.04	**50.85**
	ARI	39.15	42.49	47.00	53.51	**54.14**
	F1	67.71	72.80	75.00	78.55	**80.15**
ACM	ACC	90.45	90.59	90.90	91.67	**92.30**
	NMI	68.31	68.38	69.40	70.83	**72.53**
	ARI	73.91	74.20	74.90	76.89	**78.45**
	F1	90.42	90.58	90.80	91.66	**92.34**
CITE	ACC	65.96	68.79	69.50	70.90	**71.06**
	NMI	38.71	41.54	43.90	44.00	**55.69**
	ARI	40.17	43.79	45.50	46.47	**50.94**
	F1	63.62	62.37	64.30	63.12	**67.44**
CORA	ACC	50.70	53.70	56.87	73.58	**74.04**
	NMI	33.78	33.97	38.81	53.02	**54.39**
	ARI	25.76	24.72	29.79	**51.22**	48.47
	F1	44.13	46.27	55.92	71.48	**74.51**
AMAP	ACC	53.44	54.60	76.88	77.34	**77.50**
	NMI	44.85	49.13	69.21	64.15	**70.14**
	ARI	31.21	36.26	58.98	56.76	**60.16**
	F1	50.66	38.44	71.58	74.58	**74.70**

Table 2 Ablation study. The best results are highlighted with bold.

Dataset	Metric	AGC-AFF	w/o Attrib.	w/o Mask	w/ CMDFM	w/ FAFGC
DBLP	ACC	**80.31**	77.10	79.06	79.37	78.68
	NMI	**50.85**	45.58	48.99	50.26	49.38
	ARI	**54.14**	48.21	50.68	52.45	51.30
	F1	**80.15**	76.98	79.15	79.14	78.27
ACM	ACC	**92.30**	91.14	92.24	92.27	92.27
	NMI	**72.53**	70.08	72.46	72.38	72.38
	ARI	**78.45**	75.68	78.32	78.36	78.36
	F1	**92.34**	91.11	92.26	92.30	92.30
CITE	ACC	**71.06**	68.68	71.03	71.03	70.91
	NMI	55.69	42.62	55.68	**55.77**	55.51
	ARI	**50.94**	44.78	50.93	50.91	50.74
	F1	**67.44**	64.82	67.41	67.41	67.36
CORA	ACC	**74.04**	59.75	73.71	74.00	**74.04**
	NMI	**54.39**	46.42	53.18	54.31	54.37
	ARI	48.47	35.93	47.31	48.46	**48.58**
	F1	**74.51**	60.21	73.70	74.45	74.45
AMAP	ACC	**77.50**	74.60	75.26	67.93	68.25
	NMI	**70.14**	62.40	67.66	65.28	65.51
	ARI	**60.16**	49.92	56.20	47.25	47.67
	F1	**74.70**	69.02	72.78	67.82	68.05

4.4 Results

We perform K-means on the fused features $\mathbf{F}$ to obtain the final clustering results. The baseline methods are SDCN [5], AGCN [20], DFCN [6] and GC-SEE [16]. Each experiment is repeated for 10 times, and due to the space limitation we omit the standard deviations and only report the average values of the metrics. The results are shown in Table 1, and we can observe that the proposed AGC-AFF achieves the best performance on all the metrics across all the datasets (except the ARI metric on CORA, where AGC-AFF achieves the second-best performance[†]), which suffices to demonstrate its effectiveness.

4.5 Ablation Study

We also conduct ablation study to evaluate the effectiveness of each proposed module. Specifically, we design the following variants of AGC-AFF: 1) AGC-AFF w/o Attrib. in which ARGAE only learns to reconstruct the adjacency matrix, 2) AGC-AFF w/o Mask in which we remove the masking operation from ARGAE, 3) AGC-AFF w/ CMDFM in which we replace AFF with the fusion mechanism CMDFM proposed by [5] (in CMDFM all the nodes share the same learnable weights) and 4) AGC-AFF w/ FAFGC in which we replace AFF with the fusion mechanism FAFGC proposed by [16]. The results are shown in Table 2. From the results obtained by w/o Attrib., it can be observed that when the model is only trained to reconstruct the adjacency matrix, there is a significant decrease in its performance, indicating the effectiveness of the attribute reconstruction loss. Also, from the results obtained by w/o Mask, when the masking operation is removed from the model, there is a certain degree of decrease in its performance, demonstrating the effectiveness of the proposed masking operation. Lastly, from the results obtained by w/ CMDFM and w/ FAFGC, we can observe that the proposed AFF performs the best among the three fusion mechanisms, especially showing significant improvement on the AMAP dataset. To sum up, the model with the full ARGAE and the AFF mechanism achieves the best overall performance.

[†]The ARI metric ignores the ground truth labels of the samples and only considers whether or not any pair of samples are grouped together in the true and the predicted clusterings, and this can explain why AGC-AFF does not achieve the highest ARI but obtains the highest ACC, NMI and F1 on the CORA dataset.

5. Conclusion

In this work, we propose AGC-AFF for graph clustering, in which the graph autoencoder ARGAE simultaneously learns to reconstruct the node attributes and adjacency matrix of the graph and the fusion mechanism AFF dynamically fuses the features from AE and ARGAE based on node attention. Additionally, we design a masking operation for ARGAE, which makes the reconstruction target more difficult and helps the model learn more robust features. Through extensive experiments, we show the superiority of AGC-AFF and demonstrate the effectiveness of each proposed module.

Acknowledgments

This work is supported by the Special Fund of Huanjiang Lab, Zhejiang, China.

References

[1] X. Su, S. Xue, F. Liu, J. Wu, J. Yang, C. Zhou, W. Hu, C. Paris, S. Nepal, D. Jin, Q.Z. Sheng and P.S. Yu, "A comprehensive survey on community detection with deep learning," IEEE Transactions on Neural Networks and Learning Systems, pp.1–21, Mar. 2022.

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

[3] R. Cazabet, R. Baccour and M. Latapy, "Tracking bitcoin users activity using community detection on a network of weak signals,"

Proc. 6th Int. Conf. on Complex Networks and Their Applications, pp.166–177, Nov. 2017.

[4] J. Chen and B. Yuan, "Detecting functional modules in the yeast protein-protein interaction network," Bioinformatics, vol.22, no.18, pp.2283–2290, Sept. 2006.

[5] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu and P. Cui, "Structural deep clustering network," Proc. 29th Int. World Wide Web Conf., pp.1400–1410, Apr. 2020.

[6] W. Tu, S. Zhou, X. Liu, X. Guo, Z. Cai, E. Zhu and J. Cheng, "Deep fusion clustering network," Proc. 35th AAAI Conf. on Artificial Intelligence, pp.9978–9987, Apr. 2021.

[7] N. Mrabah, M. Bouguessa, M.F. Touati and R. Ksantini, "Rethinking graph auto-encoder models for attributed graph clustering," IEEE Transactions on Knowledge and Data Engineering, vol.35, no.9, pp.9037–9053, Sept. 2023.

[8] B. Yang, X. Fu, N.D. Sidiropoulos and M. Hong, "Towards K-means-friendly spaces: Simultaneous deep learning and clustering," Proc. 34th Int. Conf. on Machine Learning, pp.3861–3870, Aug. 2017.

[9] J. Xie, R.B. Girshick and A. Farhadi, "Unsupervised deep embedding for clustering analysis," Proc. 33rd Int. Conf. on Machine Learning, pp.478–487, June 2016.

[10] X. Guo, L. Gao, X. Liu and J. Yin, "Improved deep embedded clustering with local structure preservation," Proc. 26th Int. Joint Conf. on Artificial Intelligence, pp.1753–1759, Aug. 2017.

[11] T.N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," Proc. 5th Int. Conf. on Learning Representations, Apr. 2017.

[12] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò and Y. Bengio, "Graph attention networks," Proc. 6th Int. Conf. on Learning Representations, Apr. 2018.

[13] T.N. Kipf and M. Welling, "Variational graph auto-encoders," NIPS 2016 Workshop on Bayesian Deep Learning, Dec. 2016.

[14] J. Devlin, M. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," Proc. 2019 Conf. of the North American Chapter of the Association for Computational Linguistics, pp.4171–4186, June 2019.

[15] K. He, X. Chen, S. Xie, Y. Li, P. Dollár and R.B. Girshick, "Masked autoencoders are scalable vision learners," Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition, pp.15979–15988, June 2022.

[16] S. Ding, B. Wu, X. Xu, L. Guo and L. Ding, "Graph clustering network with structure embedding enhanced," Pattern Recognition, vol.144, pp.109833, July 2023.

[17] D. Cai, X. He and J. Han, "Locally consistent concept factorization for document clustering," IEEE Transactions on Knowledge and Data Engineering, vol.23, no.6, pp.902–913, June 2011.

[18] K.Y. Yeung and W.L. Ruzzo, "Details of the adjusted rand index and clustering algorithms, supplement to the paper an empirical study on principal component analysis for clustering gene expression data," Bioinformatics, vol.17, no.9, pp.763–774, Sept. 2001.

[19] J.O. Palacio-Niño and F. Berzal, "Evaluation metrics for unsupervised learning algorithms," arXiv preprint arXiv:1905.05667, May 2019.

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