Community Detection Based on Deep Dual Graph Autoencoder

Zhiyuan Jiang[1], Kai Xu[1], Zhixiang Wu[1], Zhenyu Wang[1](✉), and Hui Zhu[2]

[1] South China University of Technology, Guangzhou, China
{201921043738,Sekxu,202021045997}@mail.scut.edu.cn,
wangzy@scut.edu.cn

[2] College of Economics and Trade Guangdong Mechanical and Electrical Polytechnic, Guangzhou, China

Abstract. Recently, researchers try to use graph neural networks (GNNs) to solve community detection, which is a fundamental task in social network analysis. We can promote targeted products and detect abnormal users by mining the community structure in social network. In this paper, we propose the Community Detection based on Deep Dual Graph Autoencoder (CDDGA). Our model consists of the deep dual graph autoencoder module and clustering layer module. The autoencoder module can simultaneously decode the graph structure and node content. The extension path between encoder and decoder is helpful to learn higher-order structural features. We use clustering layer module to achieve better clustering performance. Finally, both modules are jointly optimized to divide communities. The experimental results show that our algorithm outperforms several state-of-the-art community detection methods.

Keyword: Community detection · Graph autoencoders · Graph neural network

1 Introduction

Community detection, also known as graph clustering, is a fundamental task in social network analysis. A common definition of community detection is to partition the nodes in the graph into some disjoint groups. We can effectively promote targeted products, detect some abnormal users and identify terrorist organizations [2] by mining the community structure in social networks.

To effectively process graph-structured data, researchers try to use GNNs to solve community detection, such as GAE [3], MGAE [3], GALA [6], ARGA [5], etc. These models decode only graph structure or node content, which will weaken the learning of the graph structure or node content. Thus Wang et al. proposed GASN [11]. To alleviate the influence of over-smoothing problem caused by the multi-layer network, Hu et al. proposed GCLN [1]. The above methods are all two-steps framework which firstly achieves the node embeddings through GNN, and then uses the K-means or Spectral clustering method for community detection. Wang et al. proposed DAEGC [9], which designs a graph clustering layer to learn community structural features.

Motivated by the above observations, we propose the deep dual graph autoencoder model CDDGA for community detection in this paper. Our contributions can be summarized as follows:

– We propose a novel deep dual graph autoencoder model CDDGA to decode graph structure and node content simultaneously.
– We stack multiple layers and use the extension path to learn higher-order neighbor features.
– We jointly optimize the dual graph autoencoder and clustering layer to achieve better clustering performance.
– Our algorithm shows state-of-the-art performance compared with other baseline techniques on community detection tasks.

2 Related Work

There are a lot of non-Euclidean data in real life, such as social networks, academic citation networks, etc. It is difficult for traditional deep learning methods to model non-Euclidean data. To effectively represent non-Euclidean data, graph neural networks have been developed.

Kipf et al. proposed GCN in 2017 [4], which can aggregate the features of first-order neighbors. However, the coefficients are the same when the GCN aggregates neighbor features, but the central node's emphasis on neighbor is different in real life. Noticing this, Veličković et al. proposed GAT [8], which uses a single-layer linear neural network to learn the attention coefficients of neighbor nodes. Although the graph attention network has achieved great results in the processing of graph data, there are still some limitations in the understanding of graph structure. Thus Xu et al. proposed GIN [14], which uses multi-layer perceptron on GCN to learn a single injection function, which has a simple structure but is very effective.

However, these methods are two-step frameworks for community detection tasks, which first use GNN to learn node embeddings, and then use K-means or Spectral to divide communities. This framework cannot integrate community structural features into node embeddings, which may lead to suboptimal clustering performance.

3 Preliminary

We consider the task of community detection on attributed graph in this paper. An attribute network $G = V, E, X$ is consists of n nodes $V = \{v_1, v_2, ..., v_n\}$ and m edges $E = \left\{e_{ij}\right\} \subseteq V \times V$. Each node v_i has a vector x_i of q dimension that describes the node content information, and the vectors of these nodes constitute the attribute matrix $X \in \mathbb{R}^{n \times q}$. The topology of G can be defined as the adjacency matrix $A = (a_{ij})_{n \times n}$, if $e_{ij} \in E$, then $a_{ij} = 1$, otherwise $a_{ij} = 0$. We consider non-overlapping community, which is defined as $C = \{C_1, C_2, ..., C_k\}$, $C_i \cap C_j = \varnothing$, $\forall i, j$. Here, C_i denotes the i-th community. Given the graph G, community detection is to divide each node v_i in G into one of k communities through a mapping function $\mathcal{F}$.

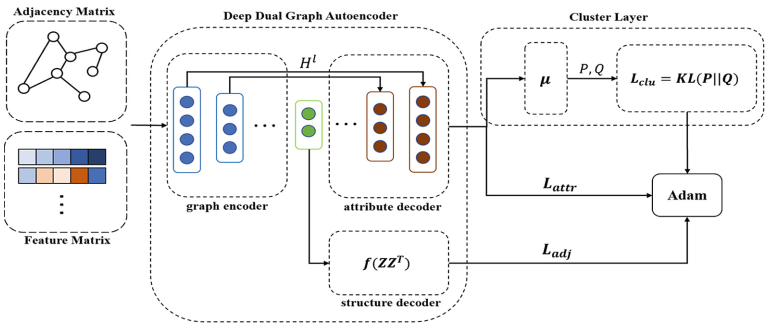

Fig. 1. Overall framework of CDDGA

4 Proposed Method

In this section, we present the Community Detection based on Deep Dual Graph Autoencoder (CDDGA) for community detection tasks. Our model consists of the deep dual graph autoencoder module and clustering layer module. The framework of the whole model is shown in Fig. 1.

4.1 Deep Dual Graph Autoencoder

The deep dual graph autoencoder module includes the graph encoder and the graph decoder. Inspired by the GAE [3], we use GCN as the layer of the graph encoder which can effectively encode both the graph structure and node content into the node embeddings. The l-th layer of the encoder is defined as follows:

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{1}$$

where $\tilde{A} = A + I_N$, A denotes the adjacency matrix of the graph, and I_N denotes the identity matrix. $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, $W^{(l)}$ is the weight matrix of the l-th layer of the encoder. $\sigma(\cdot)$ means an activation function, here we use $ReLU(\bullet) = max(0, \bullet)$. $H^{(l)} \in \mathbb{R}^{n \times d}$ denotes input feature matrix of the l-th layer of the encoder, $H^{(0)} = X$.

The graph decoder, which simultaneously reconstruct the graph structure A and node content X, consists of the graph structure reconstruction decoder and the graph attribute reconstruction decoder,

The node embeddings learned by the graph encoder part contain structure and content information, if the value of $x_u \bullet x_v$ is larger, there is a high probability that an edge is connected between node u and node v, thus we choose a simple inner product decoder as the graph structure reconstruction decoder to predict the links between nodes.

$$\widehat{A} = sigmoid\left(ZZ^T\right) \tag{2}$$

where $Z \in \mathbb{R}^{n \times d}$ denotes the node embeddings of the encoder output. We use the cross-entropy loss function to measure the difference between A and $\widehat{A}$.

$$L_{adj} = -\frac{1}{n^2}\sum_i^n \sum_j^n a_{ij}log\left(\widehat{a}_{ij}\right) \tag{3}$$

The graph attribute reconstruction decoder is symmetric with the encoder, the input feature dimension of the l layer is equal to the output feature dimension of the $L - l$ layer ($l \leq L/2$), L denotes the number of the network layer. In the experiment, $L = 8$. In order to alleviate the problem of over-smoothing, the encoder and the graph attribute reconstruction decoder are connected by the extension path, which can feed the lower-order structural information into the higher layer [1].

$$H_{expand}^{(l)} = sum\left(H^{(l)}, H^{(L-l)}\right) \tag{4}$$

Finally, we can get the reconstructed node attribute matrix $\widehat{X}$, then we use mean square error loss function to measure the difference between X and $\widehat{X}$.

$$L_{attr} = \frac{1}{n} \sum_i^n \|x_i - \hat{x}_i\|^2 \tag{5}$$

4.2 Clustering Layer

In this clustering layer, it is necessary to consider how to assign the nodes to different communities, when the embeddings of the nodes is obtained. The t-distribution can be used as a kernel function [13] to measure the similarity between the embedding h_i of node i and the embedding μ_j of cluster center j.

$$q_{ij} = \frac{\left(1+\|h_i-\mu_j\|^2/v\right)^{-\frac{v+1}{2}}}{\sum_{j'}\left(1+\|h_i-\mu_{j'}\|^2/v\right)^{-\frac{v+1}{2}}} \tag{6}$$

where v denotes the degrees of freedom in the t-distribution, we set $v = 1$. q_{ij} is a probability of assigning node i to community j, i.e. soft assignment probability. After obtaining the probability distribution $Q = \left[q_{ij}\right]$, we need to find a high-confidence probability distribution P, so that the model can optimize the node representations in the process of Q constantly approaching P. We use the method proposed by Xie [13] to calculate P by Q.

$$P_{ij} = \frac{q_{ij}^2/f_j}{\sum_{j'} q_{ij'}^2/f_{j'}} \tag{7}$$

Here, $f_j = \sum_i q_{ij}$ denotes the frequency of the soft cluster. Finally, we can use the KL divergence between the two probability distributions to obtain the following objective function.

$$L_{clu} = KL(P\|Q) = \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{8}$$

To avoid instability in the optimizing process, we consider updating Q for several iteration before updating P in our experiment.

4.3 Joint Optimization

We jointly optimize the dual graph autoencoder and the clustering layer, we define total objective function as:

$$L = L_{adj} + \beta L_{attr} + \delta L_{clu} \tag{9}$$

where L_{adj} denotes the structure reconstruction loss, L_{attr} denotes the attribute reconstruction loss and L_{clu} denotes the clustering loss. β and δ are hyperparameter. We could gain our clustering result directly from the last optimized Q:

$$c_i = \underset{j}{\text{argmax}} q_{ij} \tag{10}$$

5 Experiment

5.1 Experimental Settings

Datasets. We used three standard citation networks (Cora, Citeseer, and Pubmed) for experiments. The summary of each dataset is presented in Table 1.

Table 1. Benchmark graph datasets

Dataset	Nodes	Edges	Dims	Clusters
Cora	2708	5429	1433	7
Citeseer	3327	4732	3703	6
Pubmed	19717	44338	500	3

Baselines. We compared ten algorithms with our method in experiments.

K-means&Spectral are the most widely-used clustering algorithm.

DeepWalk [7] is a widely-used structural representation learning methods based on random walk.

M-NMF [12] is a novel modularity non-negative matrix factorization model that uses both microscopic graph structure and macroscopic community structure.

GAE&VGAE [3] combine graph convolutional network with the (variational) autoencoder to learn embeddings.

MGAE [10] introduces noise in the graph structure and node attribute.

ARGA&ARVGA [5] are adversarially regularized autoencoder, which can integrate features of graph structure and node content.

DAEGC [9] is a goal-directed graph clustering approach employing an attention network.

Metrics. In our experiment, we use three metrics to evaluate the community detection results: clustering accuracy (ACC), normalized mutual information (NMI) and adjusted rand index (ARI).

Parameter Settings. For the baseline algorithms, we carefully select the parameters for each algorithm, following the procedures in the original papers. We run the K-means algorithm 50 times to get an average value for all embedding learning baseline methods for fair comparison. For our method, we set the total number of network layers $L = 8$. For Cora and Citeseer we set the encoder neurons to be 1024–512-256–16 respectively, for Pubmed we set to 256–128-64–16. We uniformly set $\beta = 4$ and $\delta = 24$ for all the datasets. The update period of the probability distribution P is set to 3 for the stability of the optimization process. We train our model for 200 iterations using the Adam optimizer with the learning rate of 0.00001.

Table 2. Experimental results on datasets

Methods	Info	Cora			Citeseer			Pubmed		
		ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
K-means	F	0.503	0.317	0.244	0.544	0.312	0.285	0.580	0.278	0.246
Spectral	G	0.398	0.297	0.174	0.308	0.090	0.082	0.496	0.147	0.098
DeepWalk	G	0.484	0.327	0.243	0.337	0.089	0.092	0.543	0.102	0.088
M-NMF	G	0.423	0.256	0.161	0.336	0.099	0.070	0.470	0.084	0.058
GAE	F&G	0.611	0.482	0.302	0.456	0.221	0.191	0.632	0.249	0.246
VGAE	F&G	0.592	0.408	0.347	0.467	0.261	0.206	0.619	0.216	0.201
MGAE	F&G	0.681	0.489	0.436	0.669	0.416	0.425	0.593	0.282	0.248
ARGA	F&G	0.640	0.449	0.352	0.573	0.350	0.341	0.681	0.276	0.291
ARVGA	F&G	0.638	0.450	0.374	0.544	0.261	0.245	0.513	0.117	0.078
DAEGC	F&G	0.704	0.528	**0.496**	**0.693**	0.397	0.410	0.671	0.266	0.278
CDDGA	F&G	**0.714**	**0.540**	0.492	0.689	**0.429**	**0.437**	**0.686**	**0.292**	**0.305**

5.2 Experiment Result

The experimental results of different methods on different datasets are summarized in Table 2, where the values marked in bold are the best among all methods. C, S and C&S indicate if the algorithm uses only content, structure, or both content and structure information, respectively.

We can see that our method is significantly better than most of the comparison methods in evaluation metrics, which shows the effectiveness of the algorithm on the task of community detection. The methods of GAE, MGAE, ARGA and DAEGC only decode graph structure and ignore higher-order structural features. Our method decodes both the graph structure and node content, which can better fuse structure and content information. The multiple network and extension path help learn higher-order structural features in our model. Therefore, the performance of CDDGA is better than most of the baseline methods.

Table 3. The results of ablation experiment

Methods	Cora			Citeseer			Pubmed		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
CDDGA/clu	0.680	0.532	0.442	0.669	0.410	0.404	0.652	0.295	0.270
CDDGA/adj	0.683	0.539	0.464	0.657	0.383	0.401	0.649	0.279	0.262
CDDGA/path	0.608	0.438	0.368	0.582	0.300	0.302	0.603	0.176	0.173
CDDGA	**0.714**	**0.540**	**0.492**	**0.689**	**0.429**	**0.437**	**0.686**	**0.292**	**0.305**

5.3 Ablation Study

In this section, in order to analyze the effectiveness of different modules in our model, we conduct ablation experiments, The experimental results are summarized in Table 3. The comparison experiments are described as follows.

CDDGA/clu: the clustering layer module is removed.

CDDGA/adj: the graph structure reconstruction decoder is removed.

CDDGA/path: the extension path is removed.

CDDGA: complete model.

Among the three ablation experiments, CDDGA/path performed worst, because of the problem of over-smoothing. CDDGA/clu achieves great results, but it is worse than the complete model because it does not learn the features related to community structure. CDDGA/adj also lower than the complete model, because the model will focus more on the content information of nodes. According to the results, every part in the model is important.

6 Conclusion and Further Work

In this paper, we propose a novel deep dual graph autoencoder framework for community detection task. A comparison of the experimental results with several state-of-the-art algorithms validate CDDGA's community detection performance. In the future, we will try to study community detection in heterogeneous graph.

Acknowledgement. This work is funded by NSF of Guangdong (No. 2019A1515011792) ;13th five year plan project of philosophy and social sciences of Guangdong (No.GD20XYJ20); key soft science project of Guangdong electromechanical vocational and Technical College (No.YJZD20210002).

References

1. Hu, R., Pan, S., Long, G., Lu, Q., Zhu, L., Jiang, J.: Going deep: graph convolutional ladder-shape networks. Proceedings of the AAAI Conference on Artificial Intelligence **34**, 2838–2845 (2020)

2. Jin, D., et al.: A survey of community detection approaches: from statistical modeling to deep learning. IEEE Transactions on Knowledge and Data Engineering, Early Access Article (2021)
3. Kipf, T.N.,Welling, M.: Variational Graph Auto-Encoders. arXiv preprint arXiv:1611.07308 (2016)
4. Zang, Y., et al.: GISDCN: a graph-based interpolation sequential recommender with deformable convolutional network. In: International Conference on Database Systems for Advanced Applications. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-00126-0_21
5. Pan, S., Hu, R., Long, G., Jiang, J., Yao, L., Zhang, C.: Adversarially Regularized Graph Autoencoder for Graph Embedding. arXiv preprint arXiv:1802.04407 (2018)
6. Park, J., Lee, M., Chang, H.J., Lee, K., Choi, J.Y.: Symmetric graph convolutional autoencoder for unsupervised graph representation learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6519–6528 (2019)
7. Perozzi, B., Al-Rfou, R.,Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)
8. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph Attention Networks. arXiv preprint arXiv:1710.10903 (2017)
9. Wang, C., Pan, S., Hu, R., Long, G., Jiang, J., Zhang, C.: Attributed Graph Clustering: A Deep Attentional Embedding Approach. arXiv preprint arXiv:1906.06532 (2019)
10. Wang, C., Pan, S., Long, G., Zhu, X., Jiang, J.: Mgae: Marginalized graph autoencoder for graph clustering. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 889–898 (2017)
11. Wang, J., Liang, J., Yao, K., Liang, J., Wang, D.: Graph convolutional autoencoders with co-learning of graph structure and node attributes. Pattern Recogn. **121**, 108215 (2022)
12. Wang, X., Cui, P., Wang, J., Pei, J., Zhu, W., Yang, S.: Community preserving network embedding. In: Thirty-first AAAI Conference on Artificial Intelligence, **31**(1) (2017)
13. Xie, J., Girshick, R.,Farhadi, A.: Unsupervised deep embedding for clustering analysis. In: International conference on machine learning, pp. 478–487. PMLR (2016)
14. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How Powerful are Graph Neural Networks? arXiv preprint arXiv:1810.00826 (2018)