International Journal of Computer Science and Information Technology
ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 3, Number 1, Year 2024
DOI: https://doi.org/10.62051/ijcsit.v3n1.16
Journal homepage: https://wepub.org/index.php/IJCSIT/index

# Community Detection Method Based on GNN and Improved K-means

Jinxin Liu

Henan University of Science and Technology, Luoyang 471000, China
554079247@qq.com

## ABSTRACT

As the scale of networks continues to expand, their structures become increasingly complex, and the diversity of node information within these networks intensifies. Traditional community detection methods are no longer capable of meeting the growing demands. Thus, the design of a powerful community detection method with low complexity and cost is an urgent issue to be addressed. This paper proposes a community detection method based on Graph Neural Networks (GNN) and an improved K-means algorithm. Initially, the original adjacency matrix is reconstructed based on the importance of nodes, resulting in a new node similarity matrix. Subsequently, a graph autoencoder is employed to extract structural information from the network graph. The decoder part utilizes the form of vector inner product to restore the similarity matrix using the structural information from the hidden layer. The features extracted by the graph autoencoder are then utilized by the improved K-means algorithm to ultimately achieve community division. Finally, this paper summarizes the shortcomings of community detection and provides a prospect for future research directions.

## KEYWORDS

Community detection; GNN; K-means algorithm; Autoencoder

## 1. INTRODUCTION

With the rapid development of artificial intelligence technology, many complex systems in real-world scenarios can be abstracted as heterogeneous networks, such as social networks, traffic networks, and protein networks. These networks exhibit distinctive community structures. The task of community detection is dedicated to mining the community structure within these networks, which aids in a deeper understanding of the internal operating rules of complex systems and can be applied to tasks such as social recommendation and link prediction, holding significant practical implications.

Graph Neural Networks (GNN) [1] can transform complex network data into low-dimensional vector representations of nodes, offering unique advantages in handling high-dimensional network data. The embeddings learned can be used to characterize the structural information of the network [2]. For instance, the Graph Convolutional Network (GCN) extends the advantages of the Convolutional Neural Network (CNN), enabling direct convolution operations on structured network data [3]. Researchers have also used it to design community detection algorithms. In recent years, an increasing number of community detection algorithms based on graph neural networks have been developed for detecting communities in environments with complex network data structures. These have important research value in the analysis and mining of social networks, disease transmission, and other research fields [4]. Traditional community detection methods are no longer applicable for community detection in large networks, necessitating more effective methods for community mining. As an emerging technology, Graph Neural Networks are an extension and improvement of traditional

deep learning methods on graph-structured data, enabling these methods to be used on graph-structured data and filling the gap in this area.

The K-means community detection algorithm is a technique improved from the classic K-means clustering method, used to identify the community structure in networks [5]. This algorithm optimizes the similarity between nodes, dividing the network into multiple communities so that nodes within the same community have a higher connection density than nodes between different communities. In the implementation process, K nodes are first randomly selected as the initial community centers. Then, nodes are assigned to the most similar community based on their similarity to these centers (such as connection strength or the number of shared neighbors). Afterwards, each community center is updated, usually based on the average connection pattern or centrality indicators of the nodes within the community, and this process is repeated until convergence. The main advantages of the K-means community detection algorithm lie in its intuitiveness and computational efficiency. It is suitable for handling large-scale networks, can quickly converge to a stable community division, and performs well especially in networks where the community size is relatively uniform and the structure is clear. In addition, its parameter adjustment is relatively simple, mainly the choice of the number of communities K, making the algorithm easy to implement and debug.

Therefore, this paper proposes a community detection method based on GNN and an improved K-means algorithm. In summary, the main contributions of this paper are as follows:

(1) Matrix reconstruction based on node similarity. According to the importance of nodes, the original adjacency matrix is reconstructed to obtain a new node similarity matrix. The reconstructed matrix allows more nodes in the graph to be aggregated in advance, reducing the time for subsequent network processing.

(2) Graph autoencoder module. A graph autoencoder is used to learn structural information, and a two-layer graph attention network is used in the encoder part to encode the structural information of the graph and learn the low-dimensional embedding of structural information.

(3) Community detection based on the improved K-means algorithm. After the feature extraction of the graph autoencoder, the structural information in the original graph structure is effectively reduced, and then its feature matrix is used as the input of the clustering module for community division. In this module, an improved K-means algorithm is used to achieve community division.

The rest of this paper is structured as follows: Section 2 introduces the current state of research on community detection; Section 3 provides a detailed description of the proposed community detection algorithm based on graph neural networks; Finally, Section 4 summarizes the paper and looks forward to the future research direction of community detection.

## 2. RELATED WORK

The issue of community detection has received widespread attention from scholars both domestically and internationally, and various classic algorithms have been proposed. Currently, the methods for community detection mainly include traditional community detection methods and those based on deep learning.

Traditional community detection methods mainly include spectral clustering, matrix factorization, graph partitioning, density-based algorithms, statistical inference, hierarchical clustering, etc. [6]. Siemon et al. [7] used spectral analysis based on normalized Laplacian operators to discover communities in brain networks, describing the network structure at the system level rather than at individual nodes or edges. Yang et al. [8] viewed community detection as a non-negative matrix factorization task to learn the latent factors associated with node communities. On this basis, Yang et al. [9] extended the above model by extracting information about the relationship between network structure and node attributes. Zarandi et al. [10] used a hybrid approach to apply splitting and

aggregation strategies for community detection based on the CDASS algorithm, which is based on structural similarity. Wang et al. [11] performed community detection by locating structural centers and identifying communities, centers, and outliers by measuring the density of entities. Sun et al. [12] transformed the generation process of the graph and the community detection task into an inference problem for community detection.

Deep learning technology has a significant advantage in handling high-dimensional network data, thus gradually becoming one of the mainstream methods for community detection tasks. Among them, methods based on Graph Convolutional Networks (GCN) and Graph Autoencoders (GAE) are currently hot research topics [13].

GCN is a convolutional network that operates on network data, capable of integrating network topology structure information and node attribute information to learn effective node embedding representations. In 2019, Jin et al. [14] proposed JGE-CD, which uses GCN as an encoder to generate community-oriented embedding representations for community division of attribute networks. In the same year, Jin et al. [15] proposed MRFasGCN, which combines GCN with Markov Random Fields (MRF) [16] to solve the semi-supervised community detection problem in attribute networks. This method uses MRF as the last layer of the GCN model to refine the rough community division results obtained previously. GUCD proposed by He et al. [17] uses MRFasGCN as an encoder to separately reconstruct network structure information and attribute information, and further seeks potential communities through local enhancement. GAE and its variant methods are widely used in unsupervised graph representation learning tasks, and in recent years, many unsupervised community detection algorithms based on GAE have been proposed [18]. In 2017, Wang et al. [19] proposed MGAE, which designs a marginalized graph autoencoder that uses the structure and attribute information of the graph to learn effective representations suitable for community detection tasks. In 2019, Wang et al. [20] proposed a self-encoding framework DAEGC aimed at graph clustering. This framework jointly optimizes graph representation learning and graph clustering to achieve better graph clustering performance.

Therefore, we propose a community detection method based on GNN and improved K-means. This method mainly consists of three steps: First, according to the importance of nodes, the original adjacency matrix is reconstructed to obtain a new node similarity matrix. The reconstructed matrix allows more nodes in the graph to be aggregated in advance, which is beneficial to reduce the time for subsequent network processing. Second, a graph autoencoder is used to extract the structural information in the network graph, and the network information propagation model is trained in an encoding and decoding manner. The decoder part uses the form of vector inner product to restore the similarity matrix through the structural information of the hidden layer. Third, the features extracted by the graph autoencoder are used to finally achieve community division using the improved K-means algorithm.

# 3. ALGORITHM DESCRIPTION

## 3.1. Matrix Reconstruction Based on Node Similarity

This section reconstructs the original adjacency matrix based on the importance of nodes, resulting in a new node similarity matrix. The reconstructed matrix allows more nodes in the graph to be aggregated in advance, reducing the time for subsequent network processing. The matrix reconstruction method mainly consists of two parts: the first is the selection of the “core node”, that is, the first node obtained through the node importance method is used as the first node of the reconstructed matrix. The second is the selection of “neighbor nodes”. The node closest to the core node is calculated through the Euclidean distance as the second node of the reconstructed matrix, and the order of the remaining all nodes is derived in this way.

This paper uses the LeaderRank algorithm to sort nodes. LeaderRank is based on the label propagation algorithm, avoiding the disadvantages of traditional label propagation algorithms, such as low stability and accuracy, non-unique node sorting, etc., and it also has a certain tolerance for noise data. Its idea is to update the label information of nodes through the labels of neighbor nodes. A common node g is added to the network, which is connected to all other nodes in the network. At this time, the network becomes a strongly connected network with $N+1$ nodes. The algorithm first assigns 1 unit of LR value (i.e., LeaderRank value) to all nodes in the network except node g. Each node gives the average LR value to its neighbor nodes, and then updates the LR value of all nodes according to formula (1).

$$s_i(t+1) = \sum_{j=0}^{N} \frac{a_{ij}}{k_j} s_j(t) \tag{1}$$

Among them, $a_{ij}$ is the element in the adjacency matrix. If $i$ and $j$ are connected by edges, it is 1; otherwise, it is 0. $k_j$ is the degree of node $j$ in the degree matrix, and the node information is updated each time by combining the LR value of neighboring node $j$ and the probability of node $i$ randomly walking to node $j$. $t$ represents the number of iterations. At the initial state, $s_i(0)=1$ for all nodes, while $s_g(0)=0$, and iterate continuously according to equation (2) until convergence.

$$s_i = s_i(t_c) + s_g(t_c) \cdot N^{-1} \tag{2}$$

The LeaderRank algorithm can be used to obtain the node number with the highest LR value in the node set. It is labeled as a "core node" and expressed as $s_1$ using the formula. It will be the first node in the reconstructed similarity matrix. Next, use equation (3) to determine the node closest to it as the successor node, denoted as $s_2$.

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{3}$$

Where, $n$ represents the number of feature vectors in the node, and the successor node with the smallest distance from the "core node" is obtained by calculating the difference in feature vectors of the corresponding dimension between node $x$ and node $y$. According to the LeaderRank algorithm and the Euclidean distance calculation formula, the reconstruction order of all nodes can be obtained. The "core node" is the first node, and its nearest neighbor node is the second node. The subsequent nodes obtained in sequence are represented as $s_3$, ..., $s_n$. Therefore, the node order of the reconstructed similarity matrix $S$ is $S=\{s_1, s_2, \cdots, s_n\}$, and the expression of the similarity matrix s is shown in equation (4). The reconstructed similarity matrix can better feedback the importance of nodes and highlight the connectivity between nodes.

$$\boldsymbol{S} = \begin{bmatrix} \boldsymbol{s}_{l,l} & \cdots & \boldsymbol{s}_{l,n} \\ \vdots & \ddots & \vdots \\ \boldsymbol{s}_{n,l} & \cdots & \boldsymbol{s}_{n,n} \end{bmatrix}, \boldsymbol{s}_{ij} = \begin{cases} \boldsymbol{1}, \boldsymbol{e}_{ij} \in E \\ \boldsymbol{0}, \boldsymbol{e}_{ij} \notin E \end{cases} \tag{4}$$

### 3.2. Graph Autoencoder Module

This section uses a graph autoencoder to learn the structural information of the network graph, and uses a two-layer graph attention network in the encoder part to encode the structural information of the graph, learning the low-dimensional embedding of structural information. The Graph Attention

Network (GAT) uses an attention mechanism to aggregate the neighbors of nodes, adaptively assigning different weights to different neighbors, learning the hidden representation of nodes, and has strong expressive power [21].

For node $v_i$, this section uses $v_j \in \tilde{N}(v_i)$ to represent its neighbor nodes, and uses a single-layer fully connected layer to calculate the weight coefficient from $v_j$ to $v_i$, that is, the relevance of the two nodes. The calculation method is shown in formula (5).

$$h_{ij} = LeakyReLU\left(a^T\left[WX_i \,||\, WX_j\right]\right) \tag{5}$$

Among them, $X_i \in R^{d(l-1)}$ and $X_j \in R^{d(l-1)}$ represent the feature vectors of length $d(l-1)$ corresponding to nodes $v_i$ and $v_j$ in the $(l-1)$-th layer, respectively, and the weight parameter $W \in R^{d(l)\times d(l-1)}$ is used for the feature transformation of nodes. The weight parameter $a \in R^{2d(l)}$, $LeakyReLU$ is the activation function.

In order to better allocate weights, this section will normalize the calculated correlation of all neighbors using formula (6) to softmax normalization.

$$\alpha_{ij} = \mathrm{softmax}_j\left(h_{ij}\right) = \frac{\exp\left(h_{ij}\right)}{\sum_{v_k \in \tilde{N}(v_i)} \exp\left(h_{ik}\right)} \tag{6}$$

Among them, $\alpha$ is the weight coefficient, and formula (6) ensures that the sum of weight coefficients for all neighbors of the node is 1.

After calculating the weight coefficients, a weighted sum is performed based on the attention mechanism to obtain the feature vector of node $v_i$ in the $l$-th layer. The calculation method is shown in equation (7).

$$X_i^l = \boldsymbol{\sigma}\Big(\sum_{v_j \in \tilde{N}(v_i)} \boldsymbol{\alpha}_{ij} W X_j^{l-1}\Big) \tag{7}$$

This section uses a two-layer graph attention network to generate node embeddings, taking $X_i$ as input, and the resulting node embedding calculation is shown in equation (8).

$$Z_i = \boldsymbol{\sigma}\Big(\sum_{v_j \in \tilde{N}(v_i)} \alpha_{ij} W^1\big(\boldsymbol{\sigma} \sum_{v_j \in \tilde{N}(v_i)} \boldsymbol{\alpha}_{ij} W^0 X_j\big)\Big) \tag{8}$$

Where, $W_0$ represents the weight parameter of the first layer feature transformation, and $W_1$ represents the weight parameter of the second layer feature transformation.

### 3.3. Community Detection Based on Improved K-means Algorithm

After the feature extraction of the graph autoencoder, the structural information in the original graph structure is effectively reduced, and then its feature matrix is used as the input of the clustering module for community division. In this module, considering that the K-means algorithm can ensure good scalability and high processing efficiency when dealing with large-scale data, and the results are intuitive, this paper improves the K-means algorithm for node clustering and makes adaptive improvements to the selection of initial clustering centers.

First, this section uses the fused representation of nodes to solve the influence of nodes. Then, the obtained node influence is sorted in descending order, and the k nodes with the highest influence are selected as the initial clustering centers of the K-means algorithm. Then, the K-means algorithm is iteratively used until a stable community division appears. Before calculating the influence of nodes, it is necessary to consider the distance calculation of nodes. Combining the standard Euclidean distance can balance the characteristics of independence and correlation between multiple dimensions. This section uses the standardized Euclidean distance to calculate the distance between nodes $v_i$ and $v_j$, uses the optimized fusion representation of nodes as input, and the calculation of node influence is shown in formula (9).

$$P_{v_i} = \frac{1}{\sum_{v_j \in V} \sqrt{\sum_{k=1}^{n} \left( \frac{h_{ik} - h_{jk}}{s_k} \right)^2}} \tag{9}$$

Here, $P_{V_i}$ represents the influence of node $v_i$ in social networks.

## 4. CONCLUSION AND FUTURE WORK

This paper researches and proposes a community detection method based on Graph Neural Networks (GNN) and an improved K-means algorithm. Firstly, the matrix is reconstructed based on the importance of nodes. Then, a graph autoencoder is used to extract the structural information of the network graph, and the decoder part uses the form of vector inner product to restore the similarity matrix. Next, the features extracted by the graph autoencoder are input into the improved K-means algorithm based on the influence of user nodes to finally achieve community division. By summarizing the work of this paper, the following future research directions are proposed:

The network information extraction methods proposed in this paper are all applied to static networks. However, networks evolve over time, and static information extraction cannot represent the complete information of the network after evolution [22]. Therefore, in future research, the focus can be placed on how to dynamically generate and obtain network structure information. In addition, this paper uses graph neural networks for community detection, but it has not made significant improvements in time complexity. Now that graph neural networks have achieved a great expansion and have solved some problems in terms of time, in the future, methods with lower time complexity can be used for improvement to achieve community detection. The efficiency issue is also a major aspect of the algorithm, so dedication to solving efficiency will greatly promote the development and research in this field.

## REFERENCES


[1] Shchur O, Günnemann S. Overlapping community detection with graph neural networks [J]. arXiv preprint arXiv: 1909. 12201, 2019.

[2] Liu F, Xue S, Wu J, et al. Deep learning for community detection: progress, challenges and opportunities [J]. arXiv preprint arXiv: 2005. 08225, 2020.

[3] Zhang S, Tong H, Xu J, et al. Graph convolutional networks: a comprehensive review [J]. Computational Social Networks, 2019, 6(1): 1-23.

[4] He D, Song Y, Jin D, et al. Community-centric graph convolutional network for unsupervised community detection [C]//Proceedings of the twenty-ninth international conference on international joint conferences on artificial intelligence. 2021: 3515-3521.

[5] Jia S. A study on the improvement of K-means algorithm based on community discovery [C]//5th International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM 2023). IET, 2023, 2023: 312-317.

[6] Su X, Xue S, Liu F, et al. A comprehensive survey on community detection with deep learning [J]. IEEE Transactions on Neural Networks and Learning Systems, 2022.

[7] de Lange S C, de Reus M A, van den Heuvel M P. The Laplacian spectrum of neural networks [J]. Frontiers in computational neuroscience, 2014, 7: 189.

[8] Yang J, Leskovec J. Overlapping community detection at scale: a nonnegative matr-ix factorization approach [C]//Proceedings of the sixth ACM international conferen-ce on Web search and data mining. 2013: 587-596.

[9] Yang J, McAuley J, Leskovec J. Community detection in networks with node attrib-utes [C]//2013 IEEE 13th international conference on data mining. IEEE, 2013: 1151-1156.

[10] Zarandi F D, Rafsanjani M K. Community detection in complex networks using str-uctural similarity [J]. Physica A: Statistical Mechanics and its Applications, 2018, 503: 882-891.

[11] Wang X, Liu G, Li J, et al. Locating structural centers: A density-based clustering method for community detection [J]. PloS one, 2017, 12(1): e0169355.

[12] Sun F Y, Qu M, Hoffmann J, et al. vgraph: A generative model for joint community detection and node representation learning [J]. Advances in Neural Information Pro-cessing Systems, 2019, 32.

[13] Cai B, Wang Y, Zeng L, et al. Edge classification based on Convolutional Neural Networks for community detection in complex network [J]. Physica A: Statistical Mechanics and its Applications, 2020, 556: 124826.

[14] Jin D, Li B Y, Jiao PF, et al. Community detection via joint graph convolutional network embedding in attribute network [C]. Proceedings of ICANN, 2019: 594-606.

[15] Jin D, Liu ZY, Li W H, et al. Graph convolutional networks meet markov random fields: Semi-supervised community detection in attribute networks [C]. Proceedings of the AAAI Conference on Artificial Intelligence, 2019: 152-159.

[16] He D X, You X X, Feng ZY, et al. A network-specific markov random field approach to community detection [C]. Proceedings of the 32nd AAAI Conference on Artificial Intelligence, 2018: 306-313.d [45] He D X, Song Y, Jin D, et al. Community-centric graph convolutional network for unsupervised community detectionC]. Proceedings of the 29th International Joint Conference on Artificial Intelligence, 2021: 3515-3521.

[17] He D X, Song Y, Jin D, et al. Community-centric graph convolutional network for unsupervised community detection [C]. Proceedings of the 29th International Joint Conference on Artificial Intelligence, 2021: 3515-3521.

[18] Zheng Y P, Chen S Y, Zhang X N, et al. Heterogeneous-temporal graph convolutional networks: Make the community detection much better [J]. arXiv preprint arXiv: 1909.10248, 2019.

[19] Wang C, Pan SR, Long GD, et al. MGAE: Marginalized graph autoencoder for graph clustering [C]. Proceedings of the Conference on Information and Knowledge Management, 2017: 889-898.

[20] Wang C, Pan S R, Hu R Q, et al. Attributed graph clustering: A deep attentional embedding approach [C]. Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019:3670-3676.

[21] Qiu CY, Huang ZC, Xu WZ, et al. VGAER: Graph neural network reconstruction based community detection [C]. Proceedings of AAAI: DLGAAAI'22, 2022.

[22] Zhou X C, Su L T, Li X J, et al. Community detection based on unsupervised attributed network embedding [J]. Expert Systems with Applications, 2023, 213: 118937.