Community Detection based on Autoencoder Reconstruction Similarity Matrix

Yinlong Li, Qilong Han*, Jiayi Liu
College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China
*Corresponding author: hanqilong@hrbeu.edu.cn

Abstract. Community detection plays an important role in various fields. In the existing community detection algorithms, adjacency matrix is widely used in the process of community detection. However, the adjacency matrix cannot show the tightness between nodes and cannot show the characteristics of nodes. To solve this problem, we propose the concept of similarity matrix. We calculate the similarity of any two connected nodes and get the similarity matrix. Then, we use autoencoder to reconstruct the similarity matrix. The reconstructed similarity matrix can better reflect the node features. Based on the reconstructed similarity matrix, we propose a feature distance optimization formula. According to the feature distance optimization formula, we continue to optimize and reconstruct the community until we get the optimal community. Experiments on multiple real data sets show that compared with other classical efficient community detection algorithms, our proposed algorithm is more accurate in community detection.

1. Introduction

In reality, there are various complex networks composed of node and node relationships. The mining of this information is conducive to the development of enterprises and other disciplines. In addition to the small world characteristics, many real networks have a power distribution that follows the power law distribution. Community detection is an effective means of network information mining. Studies have shown that complex networks present a distinct community structure. A community is a subgraph made up of nodes in the network. The nodes in the community are closely connected, and the nodes between the communities are relatively sparse. Community detection algorithms for community detection of the network will play an important role in areas such as friend recommendation, product introduction, protein function prediction and criminal gang detection.

Many of the existing community detection algorithms involve the processing of high-dimensional adjacency matrices such as k-means and spectral clustering. However, these adjacency matrices can only reflect the connection relationship between a node and its first-order neighbors and the tightness between nodes cannot be reflected. Second, the adjacency matrix does not reflect the features of the nodes in a network structure. In view of the above problems, we proposes a community detection based on Autoencoder Reconstruction Similarity Matrix (CARS). The CARS algorithm can synthesize node tightness and node features for accurate community detection. The main contributions of this paper are as follows.

(1) Calculate the similarity of two connected nodes. Converting the network into a similar matrix makes the similar matrix not only reflect the similarity between directly connected nodes.
(2) Reconstruct the similarity matrix using the autoencoder method. The reconstructed similarity matrix is used to represent node features and community detection.

(3) Propose a feature distance optimization formula and conduct community detection based on the formula. Some experiments have been done on multiple real datasets. The experimental results show that the CARSM algorithm proposed in this paper can obtain a more accurate community structure.

The rest of this article is organized as follows. Section 2 introduces related work. Section 3 gives the formal definition and optimization formula. Section 4 proposes a CARSM algorithm. Section 5 experiments on multiple datasets to verify the validity of the CARSM algorithm. Section 6 summarizes the full paper.

2. Related Work
At present, there are many algorithms about community detection, which are roughly divided into several categories. (1) Algorithms based on modularity optimization, which mainly include greedy algorithm [1], extreme value optimization algorithm [2] and simulated annealing [3]. (2) The methods based on network topology structure, which mainly include community structure detection algorithm based on spectral analysis method [4], GN hierarchical clustering algorithm and spectral clustering algorithm [5]. (3) Integrate other disciplines for community detection, such as Wu et al. [6] regard the network as the current algorithm of the circuit. In addition, there are algorithms based on label propagation [7], such as LPA [8], HNP [9].

Graph embedding methods, such as Deepwalk [10], LE [11], GraRep [12]. The Deepwalk algorithm uses a random walk and skip-gram model to get the representation matrix of the network graph and calculate the community. The basic idea of the LE algorithm is to obtain the low-dimensional space of the adjacency matrix by Laplacian feature mapping and then perform community detection. The adjacency matrix cannot show the tightness between nodes and cannot show the features of nodes. To solve this problem, we calculate the similarity of any two connected nodes and get the similarity matrix. Then, we use autoencoder to reconstruct the similarity matrix. The reconstructed similarity matrix can better reflect the node features. Based on the reconstructed similarity matrix, we propose a feature distance optimization formula and detect the communities.

A autoencoder that can be tried to copy input to output after training is a way of neural network and common feature extraction. There is a hidden layer $h$ inside the encoder, which can generate an encoded representation input. The network can be viewed as two parts: an encoder represented by a function $h = f(x)$ and a decoder $r = g(h)$ that generates a reconstruction. If a autoencoder simply makes the output exactly equal to the input, $g(f(x)) = x$, the effect of this autoencoder is small. In order to learn the features of the data, the autoencoder output should approximate the original output instead of being completely copied.

3. Problem Statement
3.1. Definition
Given a network $G=(V,E)$, community detection is to divide a graph $G$ into $p$ disjoint subgraphs $G_i=(V_i,E_i)$, where for any $i \neq l$, $V_i \cap V_j = \phi$ and $V = \bigcup_{i}^{l}V_i$. The existing algorithm uses the adjacency matrix $A = [a_{ij}]_n$ to represent the connection relationship between nodes. The value of the corresponding element of the matrix indicates whether the edge exists. If there is an edge between $v_i$ and $v_j$, $a_{ij} = 1$. There is no edge between $v_i$ and $v_j$, $a_{ij} = 0$. However, the tightness between nodes cannot be reflected by the adjacency matrix.

**Definition 1. (Neighbor Similarity):** Given a network $G=(V,E)$ and $i,j \in V$, if there is an edge between $i$ and $j$, the neighbor node set of node $i$ is denoted as $N_i$, and the neighbor node set of node $j$ is denoted as $N_j$, the similarity of node $i$ and node $j$ is called neighbor similarity which is denoted as $s_{ij}$.
The neighbor similarity is calculated as follows.

\[ s_i = \frac{|N_i \cap N_j| + 1}{|N_i \cup N_j|} \]  

(1)

**Definition 2. (similarity matrix):** Given a network and \( G = (V, E) \), use \( S = [s_{ij}]_{n \times n} \) to represent the tightness of any two nodes in \( G \). The element \( s_{ij} \) of the matrix \( S \) is represented by the node similarity and \( S \) is called the similarity matrix.

The adjacency matrix can only reflect the topology of the network, but it does not reflect the tightness between nodes. The similarity matrix is proposed to solve the problem of node tightness. Although the similarity matrix can show the tightness of the nodes, the node features cannot be displayed. We reconstruct the similar matrix using the autoencoder method and denote the reconstructed similar matrix as \( \tilde{S} \). We set a small number of nodes in the hidden layer of the autoencoder, denoted as \( n' \) (\( n' < |V| \)). In order to achieve nonlinear dimensionality reduction of the similar matrix, the activation function of the hidden layer uses \( \tanh(x) \).

### 3.2. Feature distance optimization formula

Given a network \( G = (V, E) \), after calculating the similarity matrix \( S \), we have obtained the closeness between the nodes. On this basis, we reconstruct the similar matrix \( S \) by the autoencoder method to obtain a new similarity matrix \( \tilde{S} \). Each row of \( \tilde{S} = [\tilde{s}_i, \tilde{s}_j, \ldots, \tilde{s}_n] \) in matrix \( \tilde{S} \) represents the features of a node \( i \). Community detection will take place on the matrix \( \tilde{S} \). Given the parameter \( k \), we randomly select the nodes as the community center \( (c_1, c_2, \ldots, c_k) \) and the remaining \( n-k \) nodes assign to a community \( C_h \) \((1 \leq h \leq k)\) to get the community \( C = (C_1, C_2, \ldots, C_k) \), where \( n = |V| \).

Obviously, the community that the randomly assigned nodes is not the optimal community. In order to get an optimal community detection result, we propose a feature distance optimization formula to optimize and reconstruct the community. Community detection divides the network and makes the internal nodes of the community tightly connected and the nodes between the communities are loose. Our goal of optimizing the community is to minimize the node characteristics of all communities. We want to minimize the sum of the distances from nodes in any community \( C_h \) to community center \( c_h \).

For any node \( q \in C_h \), the formula for calculating the distance between node \( q \) and community center \( c_h \) is as follows.

\[ \text{dist} = \sum_{i=1}^{n} \left\| \tilde{s}_i - c_i \right\| \]  

(2)

where \( \tilde{s}_i \) is the row vector of the reconstructed similarity matrix corresponding to the node \( q \in C_h \).

Formula (2) is the optimization goal of community detection. We select \( k \) nodes as the community centers and assign the remaining nodes one by one to a community that minimizes the value of Formula (2). If any node \( i \in V \) connects to \( C_h \), the node is assigned to a community that minimizes formula (2). When all the points are allocated, we select a node from \( C_h \) in order as the central node and reconstruct the community until formula (2) reaches the minimum and remains unchanged. The current communities are the best communities.

### 4. Algorithm

This section mainly introduces the CARSM algorithm proposed in this paper and analyzes the algorithm. The specific process of the CARSM algorithm is as follows.

**Algorithm 1 CARSM Algorithm**

**Input:** network \( G = (V, E) \)

**Output:** community detection result \( C = (C_1, C_2, \ldots, C_k) \)
1: for each \( i, j \in V \) do
2:   if there is an edge between \( i \) and \( j \) then
3:     \( s_{ij} \leftarrow \) Calculate the neighbor similarity between \( i \) and \( j \);
4:     \( \mathbf{S} \leftarrow \) Reconstructed similarity matrix by using autoencoder;
5:   end if
6: end for
7: Select \( k \) nodes randomly as community center nodes \((c_1, c_2, \ldots, c_k)\);
8: \( H \leftarrow V \setminus \{c_1, c_2, \ldots, c_k\} \);
9: for each \( c_i \in \{c_1, c_2, \ldots, c_k\} \) do
10:   add \( c_i \) to \( C_i \);
11: end for
12: \( C \leftarrow \bigcup_{i=1}^{k} C_i \);
13: while \( H \neq 0 \)
14: for each \( i \in H \) do
15:   Sum \( \leftarrow 0 \);
16: for each \( j \in C_i \) do
17:   if \( s_{ij} \neq 0 \) then
18:     add \( i \) to \( C_i \);
19:     Sum_temp \( \leftarrow \) Calculate community tightness according to formula (5);
20:     \( C_i \) pop \( i \);
21: end if
22: if Sum_temp<Sum then
23:     Sum \( \leftarrow \) Sum_temp;
24:     add \( i \) to \( C_i \); \( H \) pop \( i \);
25: end if
26: end for
27: end for
28: update \( C \); update \( H \);
29: end while
30: Select a node in order from \( C_i \) as the central node and repeat steps 3 to 19 until the community tightness reaches the maximum and does not change;
31: Output \( C = \{C_1, C_2, \ldots, C_k\} \).

We perform an autoencoder method on \( S \) to obtain a reconstructed similarity matrix \( \mathbf{S} \) and select \( k \) nodes from it as the community center (line 1-line 7). The other nodes are assigned to the \( k \) communities according to formula (2). Finally, we continue to select new community centers and restructure communities from the current community (line 8-line 29). When the formula (2) reaches the minimum value and remains unchanged, the community \( C = \{C_1, C_2, \ldots, C_k\} \) (line 30-line 31) is output.

5. Experiment

5.1. Evaluation Index
The algorithm CARSM is executed on the zachary’s karate club (karate), the dolphin social network (dolphins) and the Polbooks. The details of the data set are shown in Table 1.

Table 1 Experimental datasets.

| Dataset | Number of node | Number of Edge | Number of community |
|---------|----------------|----------------|---------------------|
| Karate  | 34             | 78             | 2                   |
| Dolphins| 62             | 159            | 2                   |
| Polbooks| 105            | 441            | 3                   |

All experiments were done on a 3.2GHz Intel(R) Core (TM)i5-6500 PC with 8GB memory, running Windows 7. All algorithms were implemented in PyCharm 2017.2.4 (Community Edition).

The effectiveness of the algorithm will be measured from indicators such as Accuracy (Acc), Separation (Sep) and Normalized Mutual Information (NMI). The community detection result of the CSMFE algorithm is denoted as \( C = \{ C_1, C_2, \ldots, C_k \} \) and the true partition result on the data set is denoted as \( L = \{ L_1, L_2, \ldots, L_t \} \). For any \( C_i \) and \( L_j, (1 \leq i \leq k, 1 \leq j \leq t) \), let \( T_{ij} = |C_i \cap L_j| \), \( b_i = \sum_{j=1}^{t} T_{ij} \), \( d_j = \sum_{i=1}^{k} T_{ij} \).

5.1.1 Accuracy. Accuracy (Acc) are used to evaluate the accuracy of the community detection result. Accuracy is defined as Eq. (6).

\[
Acc = \frac{\sum_{i=1}^{k} \max \{T_{ij}\} \times \sum_{j=1}^{t} \max \{T_{ij}\}}{\sum_{i=1}^{k} b_i \times \sum_{j=1}^{t} |L_j|} \tag{3}
\]

5.1.2 Separation. Separation (Sep) is proposed to measure the one-to-one correspondence between a predicted community and a real community. Separation is defined as Eq. (7).

\[
Sep = \sqrt{\frac{(\sum_{i=1}^{k} \sum_{j=1}^{t}Sep_{ij})^2}{k \times t}} \tag{4}
\]

where \( Sep_{ij} = \frac{T_{ij}^3}{b_i \times d_j} \) indicates the consistency of community \( C_i \) and original community \( L_j \).

5.1.3 Normalized Mutual Information. Normalized Mutual Information (NMI) is use to detect the different between community detection results and the true partition of the network. Normalized Mutual Information is defined as Eq. (8).

\[
NMI = \frac{2 \sum_{i=1}^{k} \sum_{j=1}^{t} T_{ij} \log \frac{nT_{ij}}{b_i d_j}}{-\sum_{i=1}^{k} b_i \log \frac{b_i}{n} \sum_{j=1}^{t} d_j \log \frac{d_j}{n}} \tag{5}
\]

5.2. Experimental results and analysis
Figure 1 show that the algorithm we proposed achieve better experimental results no matter which index is use. In some datasets, the accuracy of the community detection results reached 100%. Overall, the communities we propose are found to have a high accuracy rate.

6. Conclusion
In this paper, we propose a community detection algorithm CARSM. We calculate the similarity between nodes and obtain the similarity matrix. The similarity matrix is reconstructed by autoencoder. The reconstructed similarity matrix shows the node features. We continuously optimize and restructure communities. The best community detection results optimized by feature distance optimization formula is output. The results show that compared with the classical community detection algorithm, the CARSM algorithm proposed in this paper obtains better results on multiple indicators and obtains a more accurate community structure.

Acknowledgements
This work was supported by the National Natural Science Foundation of China under Grant No.61370084 and No.61872105, and the Fundamental Research Funds for the Central Universities under Grant No.3072019CF0602, and the Project for Innovative Talents of Science and Technology of Harbin under Grant No.2016RAXXJ013, and the China Numerical Tank Project.

References.
[1] Newman M E 2004 Fast algorithm for detecting community structure in networks. Phys Rev E Stat Nonlin Soft Matter Phys 69(6 Pt 2):066133.
[2] Boettcher S and Percus A G 2010 Optimization with extremal dynamics Complexity 8(2):57-62.
[3] Guimerà, Roger, Nunes Amaral and Luis A 2005 Functional cartography of complex metabolic networks. Nature 433(7028):895-900.
[4] Pothen A, Simon H D and Liou K P 1990 Partitioning sparse matrices with eigenvectors of graphs Siam J.matrix Anal. appl 11(3):430-452.
[5] Ng A Y, Jordan M I and Weiss Y 2001 On Spectral Clustering: Analysis and an Algorithm Adv. Neural Inf. Proc. Syst. 14. MIT Press.
[6] Wu F and Huberman B A 2004 Finding communities in linear time: a physics approach European Physical Journal B Condensed Matter 38(2):331-338.
[7] Zhang C. L., Wang Y. L., Su B. Y. and Wang X. D. 2016 Multi-label propagation algorithm for overlapping community detection based on information entropy and local correlation J. Chin. Comput. Syst vol. 37, no. 8, pp. 16451650.
[8] K. Deng, W. P. Li, F. H. Yu and J. P. Zhang 2017 Overlapping community detection in complex networks based on multi kernel label propagation Journal on Communications vol. 38, no. 2, pp. 5366.
[9] H.-J. Li, Z. Bu, Z. Wang, J. Cao and Y. Shi 2018 Enhance the performance of network computation by a tunable weighting strategy, IEEE Trans. Emerg. Topics Comput. Intell. vol. 2, no. 3, pp. 214223.
[10] Perozzi B, Al-Rfou R and Skiena S 2014 Deepwalk: Online learning of social representations. 
In: Proc. of the 20th ACM SIGKDD Int’l Conf. on Knowledge detection and Data Mining. ACM Press.

[11] Belkin M and Niyogi P 2003 Laplacian eigenmaps for dimensionality reduction and data representation Neural Computation 15(6) 1373

[12] Cao S, Lu W and Xu Q 2015 Grarep: Learning graph representations with global structural information. In: Proc. of the 24th ACM Int’l Conf. on Information and Knowledge Management. ACM Press.