Robust Attributed Network Embedding Preserving Community Information

Yunfei Liu*, Zhen Liu†, Xiaodong Feng†‡, Zhongyi Li*
*School of Computer Science and Engineering, University of Electronic Science and Technology of China
†Department of Information Management, University of Electronic Science and Technology of China
‡Corresponding Author: fengxd1988@hotmail.com(Xiaodong Feng)

Abstract—Network embedding, also known as network representation, has attracted a surge of attention in data mining and machine learning community as a fundamental tool to treat network data. Most existing deep learning-based network embedding approaches focus on reconstructing the pairwise connections of micro-structure, which are easily disturbed by network anomaly or attack. Thus, to address the aforementioned challenge, we propose a novel robust framework for attributed network embedding by preserving Community Information (AnECI). Rather than using pairwise connection-based micro-structure, we try to guide the node embedding by the underlying community structure learned from data itself as an unsupervised learning, as to own stronger anti-interference ability. Specially, we put forward a new modularity function for high-order proximity and overlapped community to guide the network embedding of an attributed graph encoder. We conducted extensive experiments on node classification, anomaly detection and community detection tasks on real benchmark data sets, and the results show that AnECI is superior to the state-of-art attributed network embedding methods.

Index Terms—attributed network, graph embedding, community structure, network attack, anomaly detection

I. INTRODUCTION

Graphs or networks are ubiquitous in the real world, such as social networks, academic networks and protein networks, among which attributed networks are a potent tool to deal with data heterogeneity. In addition to the traditional plain networks that only model the interactions between nodes, a set of features for each node are also included in attributed networks. For example, in the academic citation network, citations between different articles constitute a network where each node is an article, and each node has text information, such as title and abstract. In these attributed networks, studies from social science [1] have shown that attributes of nodes can reflect and affect their community structure [2], and such insight can benefit extracting actionable knowledge from networks. Due to the complexity of high-dimensional and non-Euclidean structure of network topology and node attributes, the related tasks on attributed networks, including community detection, node clustering and node classification, bring the challenge of jointly capturing structure and feature information. To deal with this issue, network embedding, also known as network representation, has attracted a surge of attention in data mining and machine learning community recently as a fundamental tool to process networked data.

It aims to learn the low-dimensional representation of each node in the network while the network structure information is encoded. Early network embedding approaches are usually based on Laplacian eigenmaps [3], matrix factorization [4] and random walk [5].

Though these methods have limitations due to the shallow architecture, the success of these methods verifies the effectiveness of the subsequent deep learning-based methods. Also, these fundamental approaches only focus on the topological structure of the plain networks, ignoring the informative node attributes. More recently, methods that focus on deep learning on graphs have sprung up in large numbers, among which graph convolutional networks (GCN) [6] is a major advance. GCN can enhance the representation ability of network embedding results, which has great adaptability to attributed networks. Comprising a GCN encoder and a reconstruction decoder, graph autoencoder (GAE) and variational graph autoencoder (VGA) [7] are proposed to capture the highly non-linear property, which can work well for attributed network embedding and downstream tasks. Some advanced approaches are developed following this point, such as AGE [8]. Nevertheless, these methods have two major limitations.

Firstly, in essence, these network embedding methods mainly focus on the micro-structure of the network, where only the pairwise connection or similarity between each two nodes is preserved in the new embedding space [7], [8]. However, as an important mesoscopic description of network structure, community structure has been ignored to a great extent in these aforementioned methods. It is natural that most networks are composed of different communities, which are recognized as a group of nodes, and the connections within communities are dense, while the connections between communities are sparse. Community structure is one of the most remarkable characteristics of the network [9], which reveals the organizational structure and functional composition of the networks. Thus, whether the embedded representation can well reflect the community structure is a key requirement of network embedding method, which is also important for attributed networks. Different from pairwise connections of micro-structure, community structure imposes constraints on node representation at a higher structural level, that is, the representation of nodes in the same community should be more similar than those belonging to different communities [10].
between two nodes belonging to the same community, but the community structure constraint will strengthen their similarities. Therefore, the integration of community structure into network embedding can provide effective and rich information to solve the issue of data sparsity in micro-structure of pairwise connection, and can learn more discriminative node representation.

Secondly, most of the deep learning-based network embedding methods are very vulnerable to network attack [11] or node pollution [12], [13]. When the network is attacked or polluted, the individual fake or outlier edges will play a misleading role [11], and even a slight disturbance attack on the nodes will lead to a big bias. Some studies have shown that the learned representations of nodes are too sensitive to the minor change of micro-structure, such as adding or deleting an edge or node [14], and the performance of down-stream tasks, such as node classification or link prediction, will deteriorate significantly [15]. It is usually the consequence of overemphasizing the first-order proximity from the local structure and ignoring the community structure. If the unsupervised information of community membership can be considered to guide the network embedding, the results would be robust to the attack or node pollution. The community ownership of the nodes, as the global structure information, would not be easily changed by the minor local changes in network structure.

Motivated by the above observations, we propose a robust framework for attributed network embedding, named Attributed Network Embedding preserving Community Information (AnECI). Firstly, AnECI uses a graph convolutional encoder to learn the non-linear embedding of attributed networks by propagating the attribute information by graph convolution operation. Secondly, instead of pairwise connection-based model learning, a new modularity function is proposed to guide the network embedding with preserving underlying community structure learned from data itself as an unsupervised learning, which tries to improve the robustness for local network outlier or attack. Moreover, different from the conventional modularity function defined on the first-order proximity of the network and hard partition-based community structure, we develop a new modularity function for high-order proximity and the overlapped community structure. Furthermore, AnECI will reconstruct the high-order proximity by the learned node embedding vectors as the decoder, which is also different from the traditional method that usually reconstructed the first-order adjacency matrix. The high-order proximity would be more stable than first-order proximity when there is slight attack or node pollution, as it also considers more global information when computing the proximity between two nodes. So far as we know, our work is the first try to incorporate high-order proximity-based community modularity into network embedding and reconstruct high-order proximity by learned representation. We conducted extensive experiments on node classification on clean and attacked graphs, anomaly detection, and community detection tasks on real benchmark data sets, and the results show that AnECI is superior to the state-of-arts attributed network embedding methods.

The contributions of this paper are summarized as follows:

- We proposed AnECI, a robust network embedding model which enhances the robustness of embedding against adversarial attacks. To the best of our knowledge, it is the first work to investigate this challenging task.
- Instead of the conventional modularity function only focusing on the first-order proximity and hard partition-based community structure, we proposed a new modularity function that can be adapted for high-order proximity and overlapped community structure.
- Extensive experimental results demonstrate the effectiveness of proposed method on various downstream tasks.

II. RELATED WORK

A. Attributed Network Embedding

Network embedding that learns the low-dimensional embedding vectors of nodes in the network has gained much attention in the last few years, which can be traced back to the traditional graph embedding [3]. Inspired by the word embedding [16] in natural language processing, DeepWalk [5] and Node2Vec [17] are some of the first works that perform random walk on the network to generate node sequences and learn the representation by skip-gram models. For attributed network, earlier study [18] exploits matrix factorization techniques to embed nodes in attributed networks, and recently deep learning techniques are widely used for network embedding.

One group of these approaches are based on traditional deep neural network on plain data, among which, SDNE [13] and DANE [19] use the traditional autoencoder model to reconstruct data features or adjacency matrix by preserving the first and high-order network proximity. Another category of recent models proposed to learn the embeddings of nodes with graph convolution operation, demonstrating superior performance. Particularly, graph convolutional networks (GCN) [6] takes the given features as the initial representation, realizing deep embeddings of nodes by a multi-layer information propagation of graph convolution. GraphSage [20] expands it to enable inductive learning by learning an aggregation function, and DGI [21] proposes to maximize mutual information between path representations and high-level summaries of a graph.

Also, some studies combine the idea of graph convolution networks and autoencoders. For example, Graph autoencoder (GAE) and variational graph autoencoder (VGAE) [7] propose to encode the network by GCN and then decode it by inner product to reconstruct the edges by minimizing the cross-entropy loss. Following GAE (VGAE), [22] further leverages graph attention networks [23] to allocate different weights of the neighbors when aggregating information. AGE [8] proposes an adaptive graph encoder iteratively strengthening the filtered features for better node embeddings, preserving both the network structure and attribute information. The scalability of attributed network embedding on massive data also attracts some attention, and random walk-based methods have been developed [24], [25].
B. Network Embedding for Outlier Detection

The above network embedding approaches only focus on the clean network data and don’t consider the influence of outliers (nodes/edges), and thus are prone to be easily affected. Thus, recent network embedding works try to detect anomalies in networks and have achieved state-of-the-art results. A semi-supervised deep learning-based approach SEANO [26], unsupervised deep learning-based approach Dominant [12] and unsupervised matrix factorization-based approach ONE [14] are proposed for outlier detection and network embedding for attributed networks. For example, Dominant [12] explicitly encodes the graph by GCN and reconstructs the topological structure and node attributes seamlessly for network embedding. Similarly, both DONE [15] and AnomolyDAE [27] jointly learn the embedding by a structure autoencoders and an attribute autoencoders, where DONE also preserves the homophily between nodes and connects the structure embedding with the attribute embedding by minimizing the difference between them. ADONE [15] further explores the role of adversarial learning to discriminate the embeddings and minimizes the effect of outliers in a coupled way.

C. Adversarial Attack and Defense for Graph Neural Networks

It is widely accepted that deep learning models are vulnerable to adversarial attacks and Graph Neural Networks (GNNs) inevitably would suffer this issue that the slight or insignificant disturbance to the input graph may make the neural network output wrong prediction [28]. The basic idea of adversarial attacks is to disturb the graph structure and node attributes so that the graph neural network models can not correctly classify some nodes, especially adding/deleting/rewiring edges. According to different settings, there are different kinds of attacks. Evasion attacks will generate new fake samples after the training phase while poisoning attacks try to change the training samples before the training process [29]. Also the attacks can focus on mis-classifying certain target nodes or just reducing the overall classification performance, as targeted [11] or non-targeted attacks [29]. In targeted attacks, direct attacks refer to the direct manipulation on the edges or features of targeted nodes while influence attacks can only modify other nodes except the targeted nodes.

Obviously, graph adversarial attacks pose a great challenge to the robustness of GNNs, which seriously limits the applicability of GNNs in practical applications. Thus, a few works are recently proposed to improve the robustness of GNNs and prevent them from adversarial attacks. Among them, RGCN [30] adopts Gaussian distributions as the hidden embedding of nodes in each convolutional layer, instead of representing nodes as vectors; and [31] has found that netatack leads to the change of the high-rank spectrum of the graph, and it suggests to use the low-rank approximations to preprocess the graph. Pro-GNN [32] proposes to jointly learn a structural graph and a robust graph neural network model from the perturbed graph guided by the common graph properties, such as low-rank, sparse and homophily property. A comprehensive survey on adversarial attack and defense for graph neural networks can be found in [33].

It is noted that all these defense models are towards improving the robustness on semi-supervised tasks, such as node classification. Different from the aforementioned outlier-aware network embedding or network defense methods, we try to explore the community structure based on high-order proximity to guide the node representation in an unsupervised way, which enables our proposed model to show more robustness regardless of intentional adversarial attacks or unintentional outliers. As far as we know, it would be the first work on it.

III. Preliminaries and Definitions

Definition 1 Attributed Network. An attributed network is typically denoted by a graph \( G = (\mathcal{V}, \mathcal{E}, X) \), where \( \mathcal{V} = \{v_1, v_2, \ldots, v_N\} \) represents the set of \( N \) nodes (or called vertices). \( \mathcal{E} \subset \{(v_i, v_j) | v_i, v_j \in \mathcal{V}\} \) is the edge sets where each edge connects two nodes in the graph, where the total number of edges is \( M \). Depending on whether the edges are symmetric or asymmetric, the network can be unsigned or signed. \( X = \{x_1, x_2, \ldots, x_N\}^T \in \mathbb{R}^{N \times d} \) is the feature matrix, where each column vector \( x_i \in \mathbb{R}^d \) is a d-dimensional attribute vector corresponding to node \( v_i \in \mathcal{V} \), and \( x_i^j \) is the \( j \)-th entry of the attribute vector for node \( v_i \). For example, it could be the TF-IDF vector of each node if the textual description is observed.

Definition 2 First-order Proximity. Based on the edge set \( \mathcal{E} \), the first-order topological structure of the network \( G \) is usually represented by an adjacency matrix \( A = \{A_{ij}\} \in \mathbb{R}^{N \times N} \), where \( A_{ij} = 1 \), indicating there exists an edge between nodes \( v_i \) and \( v_j \). To keep the generality, the self-connections are added to the graph and thus the diagonal element in \( A \) is 1. \( D = \text{diag}(d_1, d_2, \ldots, d_N) \in \mathbb{R}^{N \times N} \) can denote the degree matrix, where each element \( d_i \) corresponding to the total number of nodes connected to the associated node \( v_i \).

Definition 3 High-order Proximity. Given the adjacency matrix \( A \), the high-order proximity [34] matrix \( \tilde{A} = \{\tilde{A}_{ij}\} \in \mathbb{R}^{N \times N} \) is defined as a polynomial function of \( A \):

\[
\tilde{A} = f(w_1 A + w_2 A^2 + \cdots + w_l A^l),
\]

where \( l \) is the order; \( w = [w_1, w_2, \ldots, w_l] \) denotes the weight vector for each item to balance the structural information of each order; \( A^l \) denotes the l-order proximity matrix, defined as: \( A^l = \prod_{i=1}^{l} A \). \( f(\cdot) \) denotes the row-wise normalization to keep the matrix \( \tilde{A} \) at a specific scale, so that \( \tilde{A}_{ij} \in [0, 1] \) can be regarded as the probability that node \( v_i \) is completely connected to \( v_j \) in the high-order space.

Definition 4 Community Structure. The community structure denotes the partition of the nodes \( \mathcal{V} \) in the network \( G \). Let \( \mathcal{C} = \{c^1, c^2, \ldots, c^K\} \) denote the set of community labels, and \( P = \{p_1, p_2, \ldots, p_N\}^T \in \mathbb{R}^{N \times K} \) represents the community membership indicator matrix. If the column vector \( p_i \) for each node is constrained as a one-hot vector, the community structure is a hard partition. \( p_i^k \) can be regarded as the probability of \( v_i \) belonging to community \( c^k \), with the
sum of each row of \( P \) being 1, which make the community structure become an overlapped partition with soft weights.

**Definition 5. Attributed Network Embedding.** The aim of the attributed network embedding is to map each nodes into a low-dimensional representation \( Z = \{ z_1, z_2, \cdots, z_N \}^T \in \mathbb{R}^{N \times h} \), where each column vector \( z_i \in \mathbb{R}^h \) denotes the learned representation for node \( v_i \). Naturally, the learned embedding matrix should preserve the community structure and high-order proximity.

The main notations used in this paper are summarized in Table I.

### IV. METHODOLOGY

#### A. Overall Framework

The framework of our proposed AnECI is shown in Figure 1, composed of three modules:

- **Attributed Network Encoder by Graph Convolutional Networks.** The graph convolutional network module serves as the network encoder to learn the non-linear latent embedding of each node.

- **Community Preservation by High-order Modularity.** To get more robust node embeddings against network anomaly or attack, the representation of nodes in the embedding space should be correlated to the community structure, which is guided by the modularity function with the assumption that nodes belonging to the same community should have high proximity.

- **High-order Network Proximity Reconstruction.** Besides the preservation of mesoscopic structure of community, the learned node embeddings can also reconstruct the high-order pair-wise proximity.

#### B. Graph Convolutional Encoder

The attributed network encoder aims to embed an attributed graph \( \mathcal{G} \) into a low-dimensional space. An effective encoder should capture the basic properties of the attributed network, which is a formidable task as there are three issues to address simultaneously, i.e., network sparsity, data non-linearity and complex modality interactions. To this end, inspired by the graph convolutional networks (GCN) [6], we propose to encode the attributed network by smoothing and propagating the attribute information on the network and a nonlinear transformation of input attributes. Thus, it can alleviate the issue of network sparsity outside the observed connections between nodes. At the same time, through multi-layer non-linear transformation, the non-linearity of data and the complex interaction of two information modalities on the attributed network are captured.

Specifically, GCN extends the convolution operation to the network data in the spectral domain, and learns a new potential representation through the spectral convolution function:

\[
H^{(l+1)} = \phi(H^{(l)}; A|W^{(l)}) = \phi(D^{-1/2}AD^{-1/2}H^{(l)}W^{(l)}), \quad (2)
\]

Here, \( \phi(\cdot) \) is the activation function, such as sigmoid and LeakyReLU. In experiment, we use \( \text{LeakyReLU}(\cdot) = \max(0, \cdot) + a \cdot \min(0, \cdot) \) as activation function with \( a = 0.01 \). \( H^{(0)} \in \mathbb{R}^{N \times h'} \) and \( H^{(l+1)} \in \mathbb{R}^{N \times h_{l+1}} \) are the corresponding input and output of the \((l+1)\)-th convolution layer. We take the feature matrix \( X \in \mathbb{R}^{N \times d} \) as the input of the first layer, i.e., \( H^{(0)} = X \); the out of the last layer is the learned embedding matrix, noted as \( Z \in \mathbb{R}^{N \times h} \).

To make the embedding of each node contain community structure information, we can infer the community membership from \( Z \) if we set the embedding dimension \( h \) as the number of communities \( |C| \). Generally, we transform the learned embedding matrix \( Z \) to soft community membership matrix \( P \) by a softmax map, so that each element is between 0 and 1, and the sum of each row is 1, as:

\[
P = \text{softmax}(Z), \quad p^k_i = \frac{e^{z^k_i}}{\sum_j e^{z^k_j}}. \quad (3)
\]

#### C. Modeling Community Structure by High-order Modularity

1) **Revisiting the Traditional Modularity Function:** As one of the most widely used approaches, the modularity maximization-based community detection is applied to model the community structure, and we will extend it for high-order proximity and overlapped community structure by firstly revisiting the traditional definition. Specifically, given a network proximity matrix \( A \), which is exactly the adjacency matrix in conventional way, the modularity function [35] is defined as:

\[
Q = \frac{1}{2M} \sum_{i,j} (A_{ij} - \frac{k_i k_j}{2M}) \sigma(c_i, c_j), \quad (4)
\]

where, \( A_{ij} \) can be understood as the observed structural information between two nodes \( v_i, v_j \), i.e., edge between nodes \( v_i \) and \( v_j \); \( k_i k_j/(2M) \) is the expected structural information between these two nodes if edges are placed randomly, where \( k_i \) denotes the degree of node \( v_i \), \( c_i, c_j \) respectively denote the community labels of nodes \( v_i, v_j \), and \( \sigma(c_i, c_j) \) indicates whether they are in the same community, that is, \( \sigma(c_i, c_j) = 1 \) if \( c_i = c_j \), 0 otherwise. Therefore, the modularity is intuitively measuring the difference between the number of edges within community and the expected number in an equivalent network with randomly placed edges. The greater the value of modularity, the more the structural information contained in the community is bigger than the expected value, which reflects that the community structure of the network is more compact.

| TABLE I |
|---|---|---|---|
| Notations | Definitions | Notations | Definitions |
| \( \mathcal{G} \) | attributed graph | \( A \) | high-order proximity matrix |
| \( V \) | node set | \( C \) | community label set |
| \( E \) | edge set | \( P \) | community membership matrix |
| \( X \) | feature matrix | \( Z \) | node embedding matrix |
| \( A \) | adjacency matrix | \( H \) | hidden layer in GCN |
| \( D \) | degree matrix | \( W \) | model parameters |
| \( N \) | number of nodes | \( h \) | size of embedding |
| \( M \) | number of edges | \( Q \) | modularity function |
| \( d \) | size of features | \( M \) | total high-order degrees |
Defining a modularity matrix \( B = \{ B_{ij} \} \in \mathbb{R}^{N \times N} \), where each element \( B_{ij} = A_{ij} - k_i k_j / (2M) \), the modularity can be re-written in matrix form [10]:

\[
Q = \frac{1}{2M} \text{tr}(P^T B P), \text{s.t. } \text{tr}(P^T P) = N,
\]

where \( \text{tr}(\cdot) \) denotes the trace of a square matrix \( \cdot \), defined by the sum of diagonal elements; \( P \in \mathbb{R}^{N \times |C|} \) is the community membership indicator matrix as in definition 4. In the traditional definition of community structure of hard-partition, each row vector \( p_i \) is a one-hot vector, meaning that only one element is 1 and all others are 0, constrained by \( \text{tr}(P^T P) = N \).

2) A Generalized Modularity Function: It is noted that the above definition of modularity in (4) only emphasizes the first-order proximity and is only for hard partition-based community structure. To extend it for generalized scenario, i.e., high-order proximity-based or overlapped community structure with soft weights, we firstly propose the following modularity function as:

\[
\hat{Q} = \frac{1}{\lambda} \sum_{i,j} \sum_{c \in C} [\gamma^o_{i,j,c} E^o_{i,j} - \gamma^e_{i,j,c} E^e_{i,j}],
\]

where \( E^o_{i,j} \) and \( E^e_{i,j} \) can be respectively understood as the generalized form of the observed and expected structural information between two nodes \( v_i, v_j \). \( \gamma^o_{i,j,c} \) and \( \gamma^e_{i,j,c} \) are the corresponding adapting factor for \( E^o_{i,j} \) and \( E^e_{i,j} \), measuring the influences of the structural information between nodes \( v_i \) and \( v_j \) for community \( c \). Specifically, they are the non-decreasing function of \( \alpha_{i,c_k} \) and \( \alpha_{j,c_k} \), which respectively denote the weight of node \( v_i \) or \( v_j \) to community \( c_k \) [36], as:

\[
\gamma^o_{i,j,c_k} = F^o(\alpha_{i,c_k}, \alpha_{j,c_k}) \in [0, 1],
\]

\[
\gamma^e_{i,j,c_k} = F^e(\alpha_{i,c_k}, \alpha_{j,c_k}) \in [0, 1],
\]

and

\[
\alpha_{i,c_k} \in [0, 1], \sum_{c_k \in C} \alpha_{i,c_k} = 1.
\]

Intuitively, \( \gamma^o_{i,j,c_k} \) and \( \gamma^e_{i,j,c_k} \) should satisfy the following two equivalent conditions:

\[
\gamma^o_{i,j,c_k} = \gamma^e_{i,j,c_k} = 0, \quad \text{IFF } (\alpha_{i,c_k} = 0 \parallel \alpha_{j,c_k} = 0);
\]

\[
\gamma^o_{i,j,c_k} > 0, \gamma^e_{i,j,c_k} > 0, \quad \text{IFF } (\alpha_{i,c_k} > 0 \& \alpha_{j,c_k} > 0).
\]

In the traditional first-order proximity-based definition of (4), \( \lambda = 2M; E^o_{i,j} \) and \( E^e_{i,j} \) exactly refer to \( A_{ij} \) and \( k_i k_j / (2M) \); \( \gamma^o_{i,j,c_k} \) and \( \gamma^e_{i,j,c_k} \) are measured by \( \sigma(\epsilon_i, \epsilon_j, k) \); \( \sigma(\epsilon_i, \epsilon_j, k) = 1 \), if nodes \( v_i \) and \( v_j \) belongs to the same community \( c_k \), 0 otherwise. In the following two subsections, we will show how to design a new \( (E^o_{i,j} , E^e_{i,j}) \) and \( (\gamma^o_{i,j,c_k} , \gamma^e_{i,j,c_k}) \) on high-order proximity-based community partition and overlapped community structure respectively.

3) Extending Modularity for High-order Proximity: For networks with community structure, many nodes in the same community are often not directly connected by edges. The phenomenon that two nodes are grouped into the same community usually owes to the high-order proximity between them as they may be directly connected to another intermediary node. Thus, the community based on the first-order proximity can only represent a simple structure, but in real life it would be more complicated. Also, simply using the community labels derived from the conventional first-order modularity function to guide the node representation learning would be easily affected by outliers or attacks, that is, not robust enough. Therefore, we propose to extend the modularity function from first-order proximity to high-order structural information.

Accordingly, the definition of \( E^o_{i,j} \) and \( E^e_{i,j} \) in (6) can be extended to high-order proximity. Since \( A^l_{i,j} \) is counting the number of different paths from node \( v_i \) to node \( v_j \) in \( l \) hops, depicting the \( l \)-th order proximity between these two nodes, the sum of \( A^l_{i,j} \) over all \( l \) is, i.e., \( A_{ij} \) can be defined as the total observed high-order structural information \( E^0_{ij} \). Analogously, \( k_i k_j / (2M) \) can represent the expected high-order structural information \( E^e_{ij} \) between two nodes, where \( k_l = \sum_{j} A_{ij} \) can be regarded as high-order structural degree of node \( v_i \) and \( M = \sum_{i,j} A_{ij} \) is the sum of all high-order structural degrees.
Thus, the new definition of modularity $\tilde{Q}$ is as follows:

$$\tilde{Q} = \frac{1}{2M} \sum_{i,j} [\tilde{A}_{ij} - \bar{k}_i\bar{k}_j/(2\hat{M})] \sigma(c_i, c_j).$$

(9)

4) Extending Modularity for Overlapped Community: In the traditional modularity function, community structure is a way of hard partition, that is, each node belongs to only one community. However, due to complex network structure, the members of communities are closely connected, which leads to some similarities between nodes and the attributes across multiple communities. On the other hand, real complex networks have never been divided into sharp sub-networks, especially those formed by social interactions. For example, actors usually belong to many different communities and participate in the activities of multiple groups at the same time.

In this section, we try to extend it to overlapped community structure with soft weights by defining $\gamma^e_{i,j,c_k}$ and $\gamma^o_{i,j,c_k}$.

Before presenting our ideas, we firstly propose the following two properties that the extended function should comply with:

- **Property 1.** To keep the generality, the modularity function defined for overlapped community structure can degenerate into the common modularity on hard partition-based community structure.

- **Property 2.** Modularity function defined for overlapped communities can imply the different likelihoods of a node belonging to different communities.

To comply with **Property 1**, the following rule should be satisfied, that is, both $\gamma^o_{i,j,c_k}$ and $\gamma^e_{i,j,c_k}$ should reach 1 when the corresponding two nodes only belong to the same community $c_k$.

$$\gamma^o_{i,j,c_k} = \gamma^e_{i,j,c_k} = 1, \quad I F \alpha_{i,c_k} = \alpha_{j,c_k} = 1.$$  \hspace{1cm} (10)

Though there are several definitions of modularity for the overlapped community, these properties are not always obeyed. For example, the definition in [37] is:

$$EQ = \frac{1}{2m} \sum_{i,j} \frac{1}{O_i O_j} \left| A_{ij} - \frac{k_i k_j}{2m} \right| \sigma(c_i, c_j),$$

(11)

where, $O_i$ is the number of communities to which node $v_i$ belongs. Obviously, it complies **Property 1**, since $O_i$ will be always 1 for hard partition-based community structure and $\tilde{Q}$ will become $Q$ in (4). However, in (11), $1/(O_i O_j)$ can be regarded as the adapting factor $\gamma$, and obviously it can not be adaptive to the overlapped community that allocates different weights for different communities of one node. Thus, $\tilde{Q}$ in (11) can not conform to **Property 2**.

Also, [36] introduced a new definition for networks with overlapped community structure, as:

$$Q^* = \frac{1}{M} \sum_{c_k \in C} \sum_{i,j} \left[ \gamma_{i,j,c_k} E_{i,j} - \frac{1}{N^2} \sum_{l=1}^N \gamma_{i,l,c_k} \sum_{l=1}^N \gamma_{l,j,c_k} E_{i,l} E_{l,j} \right].$$

(12)

According to the framework in (6), $\gamma_{i,j,c_k}$ and $\gamma_{i,j,c_k}$ can be understood as $\gamma^e_{i,j,c_k}$ and $\gamma^o_{i,j,c_k}$ respectively. However, the proposed modularity function in (12) does not satisfy the **Property 1** and we can simply prove it using proof by contradiction as follows. Supposing it satisfies the **Property 1**, given a hard partition-based community structure, when nodes $v_i$ and $v_j$ belong to the same community $c_k$, i.e., $\alpha_{i,c_k} = \alpha_{j,c_k} = 1$, we have $\gamma_{i,j,c_k} = \sum_{l=1}^N \gamma_{i,l,c_k} \sum_{l=1}^N \gamma_{l,j,c_k} = \sum_{l=1}^N \gamma_{i,l,c_k} \sum_{l=1}^N \gamma_{l,j,c_k}$, according to (10). In this case, due to the requirements in (7), $\gamma_{i,j,c_k} = 1$ must holds for all $v_i$, that is, all nodes belong to the same community $c_k$ and $|C| = 1$, which is meaningless for real community structure. In other words, if there exist more than one communities, $\gamma_{i,j,c_k}$ will never reach 1. Thus, modularity function in (12) does not satisfy the **Property 1**.

Based on the above analysis, we propose to modify the definition in (12) as: $\gamma_{i,j,c_k}^e = \gamma_{i,j,c_k}^o = F_i \alpha_{i,c_k}, \alpha_{j,c_k}$. The simple definition of $F(\cdot)$ by the product or minimum between the corresponding two weights can make both $\gamma_{i,j,c_k}^e$ and $\gamma_{i,j,c_k}^o$ conform to (8) and (10). In experiment, we empirically set $F(\alpha_{i,c_k}, \alpha_{j,c_k}) = \alpha_{i,c_k} \alpha_{j,c_k}$. Accordingly, our newly proposed modularity function is defined as:

$$\tilde{Q} = \frac{1}{2M} \sum_{c_k \in C} \sum_{i,j} \alpha_{i,c_k} \alpha_{j,c_k} [\tilde{A}_{ij} - \bar{k}_i \bar{k}_j / 2\hat{M}].$$

(13)

In matrix form, it can be rewritten as:

$$\tilde{Q} = \frac{1}{2M} tr(P^T \tilde{B} P),$$

(14)

where, $P$ is the community membership matrix obtained by the attributed network encoder as in (3), where $p_{ik}$ is exactly the corresponding weight $\alpha_{i,c_k}$; $B_{ij} = \tilde{A}_{ij} - \bar{k}_i \bar{k}_j / 2\hat{M}$. Accordingly, with the above discussion, the definition in (13) complies with both **Property 1** and **Property 2**.

D. High-order Proximity Reconstruction Decoder

In this subsection, we will show how to reconstruct the original network structure by the outputs of attribute network encoder module. Firstly, we will try to reconstruct $\hat{A}$, rather than the adjacency matrix $A$ in GA/VE [7], as the high-order proximity considers more global information when computing the proximity and is more stable and robust than first-order proximity when there is slight attack or node pollution, as discussed before. Moreover, we exploit the community membership matrix $P$ rather than embedding matrix $Z$ in GA/VE to reconstruct the graph structure, as we assume that two nodes share similar community membership are more likely to be connected in high-order space, which further enhances the nodes embedding by community structure.

Mathematically, the decoder takes the latent embedding $P$ as input and predicts the high-order proximity between each pair of nodes:

$$\hat{A}_{ij} = \text{sigmoid}(p_i^T p_j^T).$$

(15)

Accordingly, in matrix form, we can train a proximity perdition module based on the output of attributed network encoder $P$, as follows:

$$\hat{A} = \text{sigmoid}(PP^T),$$

(16)
where, $\hat{A}_{ij} = p_i p_j^T$ measures the similarity between the membership vector $p_i$ and $p_j$ corresponding to nodes $v_i$ and $v_j$, and the higher similarity between two nodes can be interpreted with a higher probability that they are connected in high-order space. Thus, $\hat{A}_{ij} \in [0,1]$ can be regarded as the estimated probability that node $v_j$ is completely connected to $v_i$ in the high-order space. Combining the previous definitions of normalized $\hat{A}_{ij}$ under (1), we can derive two binomial probability distributions, i.e., $[\hat{A}_{ij}, 1-\hat{A}_{ij}]$ and $[\hat{A}_{ij}, 1-\hat{A}_{ij}]$, where $1 - \hat{A}_{ij}$ and $1 - \hat{A}_{ij}$ respectively denotes the true or estimated probability that the corresponding two nodes are completely disconnected in the high-order space. Thus, the difference between these two distributions can be applied to detect structural anomalies or attacks on the network. In other words, if the high-order proximity between two nodes can be highly reconstructed by the decoder, the corresponding two vectors in embedding space $Z$ will well encode these structure information. Formally, the loss of high-order proximity reconstruction decoder can be defined by the sum of generalized cross-entropy on each node pair as:

$$L_R = -\sum_{i,j}(\hat{A}_{ij}\log(\hat{A}_{ij}) + (1 - \hat{A}_{ij})\log(1 - \hat{A}_{ij})).$$  

(17)

### E. Robust Attributed Network Representation

Till now, we have introduced how to encode the attributed network by a GCN module, how to preserve community structure by a new modularity function, and how to reconstruct the high-order proximity by a decoder. To jointly learn the latent representation $Z$, the total objective function of our proposed AnECI can be formulized as:

$$\min_{W} L = -\beta_1 \hat{Q} + \beta_2 L_R,$$  

(18)

where, $\beta_1$ and $\beta_2$ are the hyper-parameters to balance the contribution of community structure and high-order proximity reconstruction. By minimizing the above objective function, our proposed robust graph convolution autoencoder can iteratively approximate the input attributed network based on the latent representation of the embedding, which can be used for downstream tasks.

### V. EXPERIMENTAL SETUP

#### A. Datasets

We use four benchmark datasets for the model evaluation, including three academic citation datasets (Cora, Citeseer, Pubmed) and one blog dataset (Polblogs). Note that Polblogs does not have attribute data, and we use the unit matrix instead. We adopt the same dataset splits as in [6] on Cora, Citeseer and Pubmed. For Polblogs, we follow the experimental setting in [6] by taking 20 nodes in each class as the training set and additional 500 and 950 nodes as the validation and test sets. The basic statistics of all four datasets are shown in Table II.

#### B. Metrics and Baselines

To evaluate the performance of the proposed model on three downstream tasks, i.e., node classification with and without adversarial attacks, anomaly detection and community detection, we adopt the same widely used metrics, i.e., Accuracy (ACC), Area Under Curve (AUC), and Modularity [9] respectively. We will report the average result over 10 rounds of independent experiments on each dataset.

For unsupervised graph embedding methods, we selected ten baselines including: DeepWalk [5], LINE [38], GAE [7], VGAE [7], DGI [21] and latest methods, including: DANE [19] applying autoencoder model preserving high-order proximity, DONE/ADONE [15] minimizing the effect of outliers while generating node embeddings, AGE [8] filtering out high-frequency noises and employing an adaptive encoder to strengthen the filtered features; CFane [39] with cross fusion layers to allow information exchange and integration between two views of graph structure and attributes.

In addition, to highlight the superior performance of the proposed AnECI and its variant on downstream tasks, we select several state-of-the-arts methods on different tasks as:

- **Anomaly detection**: Dominate [12] and Anomaly-DAE [27], which both use reconstructed graph structure or attributes to detect node anomalies.
- **Community detection**: Vgraph [40] and ComE [41].

#### C. Seeding Outliers and Adversarial Attacks

For anomaly detection task, we adopt the same outlier definition as [14]. Specifically, there are three types of community outliers in an attributed network: Structural Outlier, Attribute Outlier and Combined Outlier. We follow the seeding strategy used in [14] to ensure that these outlier nodes have similar characteristics to the normal nodes in terms of structural and attribute traits so that they cannot be detected trivially.

Moreover, we evaluate the node classification performance of AnECI under three adversarial attacks: (1) **Random Attack** randomly generates fake edges and adds them into the graph, as an illustrating example of non-targeted attacks. (2) **FGA** (Fast Gradient Attack) [42] generates adversarial attack on specific nodes and aims to fool models on these target nodes against node embedding models. FGA can conduct evasion and targeted attacks. In the experiments, we only focus on direct targeted attack of FGA. (3) **NETTACK** [11] as a targeted attack that generates adversarial perturbations targeting GNNs-based model, generates attack on specific nodes and aims to fool the model on these target nodes. In the experiments, we focus on the direct poisoning attack of NETTACK since it better matches the transductive node classification setting.

### TABLE II

**STATISTICS OF THE DATASETS**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>N</th>
<th>M</th>
<th>Classes</th>
<th>d</th>
<th>Train/val/test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cora</td>
<td>2708</td>
<td>5429</td>
<td>7</td>
<td>1433</td>
<td>140/500/1000</td>
</tr>
<tr>
<td>Citeseer</td>
<td>3327</td>
<td>4732</td>
<td>6</td>
<td>3703</td>
<td>120/500/1000</td>
</tr>
<tr>
<td>Polblogs</td>
<td>1490</td>
<td>16715</td>
<td>2</td>
<td>/</td>
<td>40/500/950</td>
</tr>
<tr>
<td>Pubmed</td>
<td>19717</td>
<td>44338</td>
<td>3</td>
<td>500</td>
<td>60/500/1000</td>
</tr>
</tbody>
</table>
D. Parameter setting

Detailed parameter settings of the proposed model on different tasks are summarized in section S.I of supplementary material\(^1\). The stopping strategies for the three downstream tasks are as follows. For the node classification task, the model runs 150 epochs, and the best embedding on the validation set is selected. For the anomaly detection task, we use an early stopping strategy based on the observed modularity training loss with the patience of 20 epochs on Cora and Citeseer while 40 epochs on Polblogs and Pubmed. For the community detection task, the model was terminated after 600 epochs. In addition, for the AnomalyDAE, the hidden layer and detection task, the parameters \((\alpha, \theta, \eta)\) of AnomalyDAE were set to \((0.3, 90, 5)\) and the epochs for training were set to 180. For all other methods, we used the default parameters reported in the original papers.

All experiments are conducted on a Linux machine with 24GB memory and NVIDIA RTX 3090 GPU\(^2\).

VI. EXPERIMENTAL RESULTS

In this section, we will report the detailed results on different tasks, the ablation experiments and the efficiency analysis of our model.

A. Node Classification on Clean Datasets

For all unsupervised graph embedding methods, we train a logistic regression classifier with node embeddings as input features on the training set and report the average accuracy (and standard deviation) on the test set. As shown in Table III, AnECI performs the best on three datasets and slightly behind GAT and DGI on Citeseer. More specifically, compared to unsupervised methods, AnECI is able to improve the accuracy against the best baseline DGI by a margin of 9.4% and 2.4% on Polblogs and Pubmed, respectively. These improvements indicate that preserving underlying community information is beneficial. It is surprising that the results of AnECI even exceed those of the semi-supervised models in most cases. We assume that these encouraging results stem from the fact that AnECI indirectly allows each node to fit the underlying community structure by strengthening relationships among nodes in the same community and weakening relationships of nodes in the different communities.

B. Against Adversarial Attacks

In this section, We first perform an analytical experiment to explain why AnECI is robust to attacks. Then we propose a denoising variant of AnECI based on the analytical experiment. Finally, we will report our defense performance under three kinds of adversarial attacks.


2Datasets and implementation of our method is available in https://github.com/Gmrylbx/AnECI.

1) Robustness study: We firstly conduct an empirical analysis to verify that the proposed approach can effectively reduce the impact of adversarial attacks and produce more robust embeddings. We choose LINE, GAE, DGI and AnECI for comparison experiments and report the defense scores of all four models under random attack at different perturbation rates. The defense score is defined as follows.

(i) Given a graph \(\mathcal{G} = (V, E, X)\), for a random attack at a perturbation rate \(0 < \delta < 1\), we mark the attack (a set of fake edges) as \(\mathcal{E}^*\), where we have \(\mathcal{E}^* \cap \mathcal{E} = \phi\) and \(|\mathcal{E}^*| = \delta |\mathcal{E}|\). Then, we generate the node embedding matrix \(Z \in R^{N \times h}\) on the attacked graph.

(ii) For each edge \(e_{i,j} \in \mathcal{E}^* \cup \mathcal{E}\), we calculate its anomaly score \(s(e_{i,j}) = 1 - (z^T_i z_j / (||z_i|| ||z_j||))\) where \(i\) and \(j\) are the two endpoints of the edge. A higher anomaly score means that the edge \(e_{i,j}\) has less influence on the embedded vectors \(z_i\) and \(z_j\). Then, we define the defense score as \(DS(\delta) = \sum_{e \in \mathcal{E}^*} s(e) / (\delta \sum_{e \in \mathcal{E}} s(e))\). It means the ratio of the average anomaly score of the fake edges to that of the normal edges. A higher defense score indicates that the node embeddings are less affected by the attacking edges and thus more robust.

We set the perturbation rate \(\delta\) increasing from 0 to 0.5 with an interval of 0.02 and report the corresponding defense scores in Fig. 2. It shows that our model has the overwhelming highest defense scores over all perturbation rates, due to the preservation of community information for unsupervised model learning. It implies that community is a relatively stable structure and can not be affected easily by the fake edge attacks, and thus the un-supervised learning signal can remain robust in noisy environments. However, the skip-gram based graph embedding such as LINE, on the other hand, has an emphasis on maintaining the first-order proximity of the embedding, and DGI and GAE both consider heavily the pairwise connections. Thus, the fake edges imposed the same effect on the model learning as the normal ones did.

Fig. 2. Defense score under random attack of different methods on Cora. Due to space limitation, we here only show the results on Cora, and the results on other three dataset refer to Section S.I in supplementary material.
resulting in the learning process being misled by the structural perturbation.

2) A denoising variant of AnECI: Based on the robustness analysis, we further propose a two-stage denoising variant, i.e., the first stage refers to network denoising and the second stage is used for network embedding. Note that in both phases, we employ the same parameters and training process to generate node embeddings. The denoising variant is described as Algorithm 1.

**Algorithm 1 AnECI+**

**Input:**
Adjacency matrix $A$, feature matrix $X$, drop ratio $\rho$.

**Output:**
Embedding matrix $Z$;

**/Denoising phase:**
1. Obtain $Z$ by running AnECI model.
2. Calculate anomaly score $s(e_{i,j}) = 1 - (z_i^T z_j \|z_i\|\|z_j\|)$ for each edge and ranking the scores in descending order.
3. Remove the edges with scores ranked in the top $\rho$ to the original network and obtain the denoising adjacency matrix $A'$.

**/Embedding phase:**
4. Let $A'$ be input and generate new embedding $Z$ by running AnECI model again.

It appears difficult to determine the value of drop ratio $\rho$ without knowing the scale of attack. Empirically, drop ratio should be proportional to the scale of attack, so we use the average defense score to represent the scale of attack, and set $\rho = \psi(\sum_{e \in E, e \neq e^*} s(e))/(|E^*| + |E|)$, where $\psi(\cdot)$ is an incremental function used to smooth the drop ratio. We define $\psi(x) = \gamma/(1 + exp(\alpha(x - \beta)))$ by which $\gamma$ determines the mapping of the drop ratio from $[0, 1]$ to $[0, \gamma]$ to ensure the denoising process maintain the basic community structure of networks. $\alpha, \beta$ determine the degree of smoothing of the drop ratio. In the experiments, we fix $\beta = 0.5, \gamma = 0.75$. For $\alpha$, it is set as 12 for Pubmed and 5 for other three datasets on NETTACK; on FGA, we fix $\alpha$ as 4, 2, 18, 3 for Cora, Citeseer, Polblogs and Pubmed, respectively; on random attack it is 2.2, 2.4, and 4 respectively for these four datasets.

3) Defense performance: To further validate the robustness of the AnECI model comprehensively, we have carefully selected three adversarial attacks. The FGA attack and NETTACK are both targeted attack methods, which specifically target node embedding models and GNNs-based models, respectively. Moreover, we use Random Attack as an illustrating example of the non-targeted attack method. Specifically, we focus on performing poisoning attacks, i.e. model is trained after the graph has been attacked.

**Against Targeted Adversarial Attack**. We then evaluate the classification performance against targeted adversarial attacks. For FGA attack, we retrain the un-supervised models on the perturbed graph, and for NETTACK, both un-supervised and semi-supervised models will be retrained on the perturbed graph. Then, we test the node classification accuracy of different models on the targeted nodes to validate whether they can defend the attacks. The nodes in test set with degree larger than 10 is set as target nodes following [30], and the number of perturbations imposed on each targeted node varies from 1 to 5. We adopt all the default parameter settings of FGA and NETTACK reported in the original papers [42], [11]. The results of node classification under the attacks of NETTACK and FGA are respectively illustrated in Fig. 3 and Fig. 4.

Fig. 3 shows that our methods AnECI and AnECI+ perform better than other network embedding methods under targeted attacks in most cases. For instance, on Polblogs dataset with 5 perturbations per targeted node, AnECI+ beats against DGI by 72% and other embedding methods by at least 16%. However, our method does not perform the best on the PubMed dataset with 5 perturbations per targeted node. We assume that when the graph is very sparse, the community structure becomes less obvious and more fragile which is easy to be damaged. The results in Fig. 4 also show similar cases on FGA attack defense, that is, our method again consistently achieves better results compared to all the baselines on Cora, Citeseer and Polblogs, demonstrating the robustness of our method under targeted attacks. We particularly note that under NETTACK, the defense performance of semi-supervised methods lags behind much un-supervised methods in most cases. We think the reason is probably that NETTACK is mainly designed for attacking semi-supervised model, although it has good transferability for un-supervised model.

**Against Non-targeted Adversarial Attack**. We continue to evaluate node classification against non-targeted adversarial attacks, i.e. perturbations that aim to reduce the model's overall classification accuracy. We use Random Attack as an illustrating example of the non-targeted attack method. Again, we focus on performing poisoning attacks, i.e. noisy edges are randomly added into the graph. We train different defense

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Semi-supervised</th>
<th>Un-supervised</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GCN</td>
<td>RGCN</td>
</tr>
<tr>
<td>Cora</td>
<td>81.5±0.5</td>
<td>81.7±0.2</td>
</tr>
<tr>
<td>Citeseer</td>
<td>70.3±0.3</td>
<td>70.8±0.5</td>
</tr>
<tr>
<td>Polblogs</td>
<td>80.6±1.6</td>
<td>83.7±0.2</td>
</tr>
<tr>
<td>Pubmed</td>
<td>79.0±0.3</td>
<td>78.7±0.2</td>
</tr>
</tbody>
</table>
models on the poisoned graph, and evaluate the classification accuracy on the test set. The ratio of number of noisy edges to clean edges varies from 0 to 50\% with an interval of 10\% and the experimental results are reported in Fig. 5. It shows that AnECI and AnECI+ consistently outperform all baselines on Cora, Citeseer and Polblogs. For the sparse graph Pubmed, the relatively poor results of AnECI and AnECI+ are likely to be caused by the same reason mentioned before, i.e. community structure can be more easily destroyed on the very sparse graph. Accordingly, we can conclude that our methods are able to effectively defend various types of adversarial attacks in an unsupervised way. This is desired in practice since robust embedding using graph structure without labels.

C. Anomaly Detection

In the anomaly detection task, we totally implant 5\% of the anomalous nodes for each graph. After obtaining the embedding $z_i$ of the node $i$, let $p_i = \text{softmax}(z_i)$, where $p_i \in R^{|C|}$ represents the community membership vector. Following [43], we calculate the anomaly score for $p_i$ as follows,

$$A\text{Score}(i) = \sum_{k=1}^{|C|} \frac{p_k^i}{p_i}$$

For anomaly detection methods (Dominate, DONE, ADONE, and AnomalyDAE), we use their default anomaly scoring methods. For other embedding methods that do not explicitly
Fig. 6. Anomaly detection results (AUC) of AnECI and baselines. Note that ‘S’, ‘A’ and ‘S&A’ respectively stand for nodes with structural anomalies, attribute anomalies and with combined anomalies; ‘Mix’ stands for each kind of the above anomalous nodes implanted by 1/3.

give anomaly detection schemes, we use the isolated forest algorithm [44] to obtain the anomaly scores from the embeddings. Finally, we report the average AUCs for all compared methods, shown in Fig. 6, indicating that AnECI performs the best on the four datasets in terms of four kinds of anomalies, except for a few cases.

D. Community Detection

We compute the community membership vector \( p_i = \text{softmax}(z_i) \) given the embedding \( z_i \) of node \( i \). Node \( v_i \) is assigned into the community \( c^k \) only if \( k = \arg \max_j p_{ij} \), i.e., node \( v_i \) is allocated into the one with the highest probability. After assigning each node a community label in this way, we calculate the traditional modularity in (4) of community partition as an evaluation metric. For other embedding methods, we employ Kmeans++ [45] to cluster the node embedding vectors and also use modularity to evaluate the clustering results. It is noted that to ensure fairness when comparing with some methods, attributed data is replaced by unit matrix when running AnECI for community detection task, since Vgraph and ComeE only use this graph structure data. As shown in Fig. 7, compared to other baselines, AnECI performs the best on three out of the four tested networks. On polblogs, it is only behind DGI.

E. Ablation Study

To demonstrate the effectiveness of the key modules deployed in the proposed framework, we perform several ablation experiments on Cora dataset.

1) Effects of Different Modules: Results are shown in Table IV, where, ‘Raw feature’ refers to directly using feature vector as embedding, ‘+Encoder’ refers to using encoder alone to generate node embeddings, ‘+Modularity’ denotes further deploying the modularity optimization, and ‘Full model’ refers to the complete model, AnECI. Results in Table IV demonstrate that the performance on all three tasks have gotten improved with the increasing integration of the key modules, indicating that these modules truly play a positive role in promoting the model.

2) Visualization: To intuitively show the learned node embeddings, we visualize the node representations in 2D space using t-SNE [46] algorithm. As shown in Fig. 8, each sub-figure corresponds to a variant in the ablation experiment.
illustrated, the graph encoder acts as a Laplacian smoothing to close the distance between similar nodes based on the original features. Adding modularity constraints alone will induce nodes to form more communities. However, optimizing only the modularity loss will make the learned community structure not coincide with the real graph structure, i.e., too much emphasis is placed on optimized modularity at the expense of the graph structure. Reconstruction loss can guide the learning of community structure, making the learned community structure match the real graph structure. Adding reconstruction loss, node representations are more clustered by similar nodes with fewer overlapping regions compared to the variant with only modularity, which represents that the model has captured the real community partition.

3) $\tilde{Q}$ v.s. $Q$: Moreover, we show whether the newly proposed modularity function based on high-order proximity and overlapped community makes sense.

**Effect of different hops.** Firstly, we report how the node classification accuracy of proposed AnECI on attacked networks changes for modularity function based on proximity of different orders and first-order proximity, shown in Fig. 9-(a). The phenomenon that the highest accuracy is obtained when adopting high-order proximity (the optimal number of hops is bigger than 1) reveals the advantage of high-order proximity-based modularity function on the robustness against the affects by network attack.

**Overlapped community v.s. hard partition.** Also, we empirically show whether the overlapped community structure-driven modularity function makes differences. Since the constraint $\text{tr}(P^TP) = N$ in (14) forces the community structure to be of hard partition, we define an index to measure how the learned community structure is close to hard partition, as $\text{Rigidity} = \text{tr}(P^TP)/N$. Thus, we show how it changes during the optimization process. As shown in Fig. 9-(b), with the increasing of the modularity $\tilde{Q}$ when minimizing the loss in (18), the community structure is being close to hard-partition (the Rigidity is being close to 1). However, the classification accuracy on test nodes (Cora dataset) reaches its peak when it is overlapped community with Rigidity < 1 (the black triangle in the curve). At last, the Rigidity reaches 1, implying a hard partition, but the classification accuracy get lower (the lowest point is marked as the red triangle). This observation thus indicates the superiority of overlapped community structure to hard partition.

In conclusion, the newly proposed modularity function $\tilde{Q}$ is superior to the conventional one. More importantly, it is of higher adaptability, which can always induce the best underlying community structure for graphs with different characteristics.

**TABLE V**

<table>
<thead>
<tr>
<th>Method</th>
<th>Cora</th>
<th>Citeseer</th>
<th>Polblogs</th>
<th>Pubmed</th>
<th>Cora</th>
<th>Citeseer</th>
<th>Polblogs</th>
<th>Pubmed</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepWalk</td>
<td>0.19</td>
<td>0.21</td>
<td>0.15</td>
<td>0.09</td>
<td>10.4</td>
<td>21.0</td>
<td>13.3</td>
<td>90.5</td>
</tr>
<tr>
<td>LINE</td>
<td>2.11</td>
<td>1.84</td>
<td>2.04</td>
<td>1.00</td>
<td>13.9</td>
<td>31.1</td>
<td>102.2</td>
<td>90.5</td>
</tr>
<tr>
<td>VGAE*</td>
<td>0.03</td>
<td>0.05</td>
<td>0.01</td>
<td>1.58</td>
<td>9.1</td>
<td>11.1</td>
<td>2.7</td>
<td>321</td>
</tr>
<tr>
<td>DGI</td>
<td>0.02</td>
<td>0.02</td>
<td>0.01</td>
<td>0.01</td>
<td>6.3</td>
<td>5.1</td>
<td>2.7</td>
<td>68.8</td>
</tr>
<tr>
<td>DANE</td>
<td>0.05</td>
<td>0.10</td>
<td>0.02</td>
<td>2.25</td>
<td>360</td>
<td>417</td>
<td>382</td>
<td>2355</td>
</tr>
<tr>
<td>DONE</td>
<td>0.01</td>
<td>0.02</td>
<td>0.01</td>
<td>0.37</td>
<td>7.1</td>
<td>3.4</td>
<td>7.0</td>
<td>146</td>
</tr>
<tr>
<td>AGF</td>
<td>0.01</td>
<td>0.02</td>
<td>0.01</td>
<td>0.37</td>
<td>4.8</td>
<td>9.7</td>
<td>2.0</td>
<td>146</td>
</tr>
<tr>
<td>CFane</td>
<td>0.22</td>
<td>1.43</td>
<td>2.94</td>
<td>8.00</td>
<td>380</td>
<td>413</td>
<td>479</td>
<td>1360</td>
</tr>
<tr>
<td>AnECI</td>
<td>0.01</td>
<td>0.02</td>
<td>0.01</td>
<td>0.42</td>
<td>2.59</td>
<td>3.2</td>
<td>1.83</td>
<td>62.7</td>
</tr>
</tbody>
</table>

**F. Running Time Comparison**

To show the efficiency of our model, we compare the running time of different baseline models on the four clean datasets for node classification, shown in Table V. It demonstrates that the running time of proposed AnECI is close to or even less than other GCN-based methods, including VGAE/GAE, DGI and AGE, and much less than Deepwalk, LINE, DANE, DONE/ADONE and CFane. These results thus illustrate the superiority of AnECI on efficiency.

**VII. CONCLUSIONS**

To alleviate the effects of network outliers and defense against network attack on network embedding, we propose a robust **Attributed Network Embedding preserving Community Information** (AnECI). Instead of pairwise connection, AnECI proposes to guide the network embedding of GCN encoder by a modularity function with preserving community structure information determined by the embedding itself as an unsupervised way. It tries to improve the robustness for usual local network outlier or attack, as it considers the global community structure in an unsupervised way. Also, AnECI reconstructs the high-order proximity, which is more robust than first-order proximity when there is slight attack or node pollution, as it considers more global information. Extensive experiments were conducted on four real datasets for three graph mining tasks, and the results show a remarkable advance of AnECI over state-of-arts methods. In the future, we try to exploit more properties to further improve the robustness of network embedding against network attacks or outliers, and improve the scalability on the larger dataset by sampling and learning aggregation function [20] instead of full graph Laplacian propagation.