Community-Centric Graph Convolutional Network for Unsupervised Community Detection

Dongxiao He\textsuperscript{1}, Yue Song\textsuperscript{1}, Di Jin\textsuperscript{1,*}, Zhiyong Feng\textsuperscript{1}, Binbin Zhang\textsuperscript{1}, Zhizhi Yu\textsuperscript{1} and Weixiong Zhang\textsuperscript{2}

\textsuperscript{1}College of Intelligence and Computing, Tianjin University, Tianjin 300350, China
\textsuperscript{2}Department of Computer Science and Engineering, Washington University, St. Louis, MO 63130, USA
{hedongxiao, sy717389667, jindi, zyfeng}@tju.edu.cn, weixiong.zhang@wustl.edu

Abstract

Community detection, aiming at partitioning a network into multiple substructures, is practically important. Graph convolutional network (GCN), a new deep-learning technique, has recently been developed for community detection. Markov Random Fields (MRF) has been combined with GCN in the MRFasGCN method to improve accuracy. However, the existing GCN community-finding methods are semi-supervised, even though community finding is essentially an unsupervised learning problem. We developed a new GCN approach for unsupervised community detection under the framework of Autoencoder. We cast MRFasGCN as an encoder and then derived node community membership in the hidden layer of the encoder. We introduced a community-centric dual decoder to reconstruct network structures and node attributes separately in an unsupervised fashion, for faithful community detection in the input space. We designed a scheme of local enhancement to accommodate nodes to have more common neighbors and similar attributes with similar community memberships. Experimental results on real networks showed that our new method outperformed the best existing methods, showing the effectiveness of the novel decoding mechanism for generating links and attributes together over the commonly used methods for reconstructing links alone.

1 Introduction

Real-world systems often appear in the form of networks. Examples include social networks, power grid and world trade networks. Real networks have modular structures or communities [Fortunato, 2010], which are densely connected subgraphs with nodes of close relationships and similar properties. Identification of network modular structures is an effective means to understanding the underlying organizational principles and functions of the system that the network represents. Community detection has been an active area of research, as surveyed in [Falih et al., 2018; Fortunato, 2010; Fortunato and Hric, 2016], and many community detection methods have been proposed, including that based on statistical modeling [Chen et al., 2018], modularity optimization [Yang et al., 2016], matrix factorization [Wang et al., 2017].

Deep learning has recently been adopted in network analysis [Kipf and Welling, 2017; Yang et al., 2016; Jin et al., 2018; Monti et al., 2018; Pan et al., 2018]. In particular, Graph Convolutional Networks (GCN) has attracted a great deal of attention lately due to its success on supervised and semi-supervised classification of nodes in a graph [Li et al., 2018; Kipf and Welling, 2017] which can be adopted for community detection. Of particular relevance to the current study is MRFasGCN [Jin et al., 2019], a state-of-the-art GCN-based semi-supervised community detection method, which incorporates a Markov Random Fields (MRF) modeling of communities in the GCN framework.

Community detection is in essence an unsupervised learning problem. Real-world networks are typically unique – the training data from one network can be hardly used adequately for another network. As a corollary to this observation, when finding communities in a network, the only data available for analysis are the information on the network itself. A scheme of semi-supervised learning, such as the MRFasGCN method, can apply in such a way that partial label information on some of the nodes in a given network can be used to predict the community identities of the remaining unlabeled nodes in the same network. However, for most application problems, even such partial training data are costly and arduous to gather.

Therefore, to advance the state-of-the-art of community detection, it is of great technical significance and paramount practical importance to develop GCN-based algorithm for unsupervised community detection by exploiting the great power of automatic feature learning and effective optimization that deep learning can offer. It also remains to be seen if an end-to-end deep learning approach can outperform the existing methods that use statistical methods and other machine learning techniques for community finding.

We developed a novel GCN-based approach for Unsupervised Community Detection in attribute networks, referred to as GUCD. In this method, we incorporated network modeling method of MRFasGCN in the Autoencoder framework and introduced a special neural network architecture suitable for learning network communities and node semantics at the same time. We further introduced a local enhancement to the...
latent communities, i.e., we first constructed an aggregated graph combining the information of both topology and attributes, and then made each pair of connected nodes in this graph have similar community distributions. We conducted experiments on some real networks to compare the new approach with the best existing methods and analyzed the features of the new approach. To our best knowledge, GUCD is the first GCN method for unsupervised community finding.

2 Preliminaries

We first introduce some notations and define the problem of community detection, and then discuss MRFasGCN [Jin et al., 2019] (a GCN based semi-supervised community detection method) which serve as the bases of our new approach.

2.1 Notations and Problem Definition

An undirected and attributed network is represented as a graph $G = (V, E, W)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes, $E$ is the set of edges with $e_{ij} = (v_i, v_j) \in E$ if an edge exists between nodes $v_i$ and $v_j$, and $W = \{w_1, \ldots, w_m\}$ is the set of node attributes. The topological structure of $G$ is represented by an $n \times n$ adjacency matrix $A = (a_{ij})_{n \times n}$, where $a_{ij} = 1$ if $e_{ij} \in E$, or 0 otherwise. An $n \times m$ attribute matrix $X$ is used to denote the attributes of nodes, in that node $v_i$ has a community identity or label $c_i \in L = \{1, 2, \ldots, K\}$.

2.2 MRFasGCN

MRFasGCN [Jin et al., 2019] infers a partition of nodes in a network in two main parts. First, they take the original two convolutional layers of GCN [Kipf and Welling, 2017], that is

$$X^{(2)} = \text{softmax} \left( \tilde{A} \text{ReLU} \left( \tilde{A}XH^{(0)} \right) H^{(1)} \right)$$  
(1)

as the first two layers of the neural network to infer an initial assignment of node labels ($X^{(2)}$), where $\tilde{A} = D^{-1/2}A\tilde{D}^{-1/2}$ ($\tilde{A} = A + I_n$ and $\tilde{D}$ is a diagonal matrix with $\tilde{d}_{ii} = \sum_j \tilde{a}_{ij}$) captures network topology. $X$ is the attribute matrix, and $H^{(0)}$ and $H^{(1)}$ are weight parameters of the two convolutional layers to be trained.

MRFasGCN then incorporates a community-oriented MRF as the third convolutional layer in the GCN. The central piece of this MRF model is an energy function, $E(C|A, X)$, with two parts: the sum of unary potentials over all nodes and the sum of pairwise potentials over all edges:

$$E(C|A, X) = \alpha \sum_i \phi_u(c_i) + (1 - \alpha) \sum_{i \neq j} \phi_p(c_i, c_j)$$

$$= \alpha \sum_i -p(c_i) + (1 - \alpha) \sum_{i \neq j} \mu(c_i, c_j)\tau(v_i, v_j)$$  
(2)

where $\alpha$ is a parameter for balancing the unary and pairwise potentials. The unary potential $\phi_u(c_i) = -p(c_i)$ measures the cost for node $v_i$ taking label $c_i$, while $p(c_i) = x^{(2)}_{i,c_i}$ is the probability that $v_i$ has label $c_i$, derived from GCN in Eq. (1). The pairwise potential $\phi_p(c_i, c_j) = \mu(c_i, c_j)\tau(v_i, v_j)$ measures the cost for assigning labels $c_i$ and $c_j$ to nodes $v_i$ and $v_j$, where $\mu(c_i, c_j)$ denotes the semantic similarities between communities $c_i$ and $c_j$, and $\tau(v_i, v_j)$ the attribute similarity or consistency between nodes $v_i$ and $v_j$, defined as:

$$\mu(c_i, c_j) = (1 - \delta(c_i, c_j))h^{(2)}_{c_icj}$$  
(3)

$$\tau(v_i, v_j) = \beta * \xi(v_i, v_j) + (1 - \beta) * R_i(\zeta(v_i, v_j))$$  
(4)

where $h^{(2)}_{c_icj}$ is the parameter to be learned, $\delta(c_i, c_j) = 1$ if $c_i = c_j$, or 0 otherwise, $\xi(v_i, v_j) = d_id_j/2e - a_{ij}$ ($d_i$ is the degree of node $v_i$ and $e$ the number of edges), and $\beta$ is a tradeoff parameter that balances topology and attributes. $\zeta(v_i, v_j)$ is defined by using the cosine similarity between the attribute vectors of nodes $v_i$ and $v_j$, and then an asymmetric regularization is used to balance the difference of the sum of similarity on every node, i.e., $R_i(\zeta(v_i, v_j)) = \zeta(v_i, v_j) / \sum_{j=1}^{m} \zeta(v_i, v_j)$.

However, minimizing the above energy function in order to yield the most probable community partition for a given network is intractable, since the pairwise potentials are defined over a complete graph rather than the sparse network. Thus, a mean filed approximation is adopted to approximate the exact distribution $P(C|A, X)$. The updating procedure for this approximation has four steps: 1) initialization, 2) message passing, 3) adding unary potentials and 4) normalization. Following these steps, MRFasGCN can transform the MRF’s inference into a layer of convolutional process (that is compatible with GCN of Eq. (1)), defined as:

$$Z = \text{softmax} \left( X^{(2)} - \Upsilon X^{(2)} H^{(2)} \right)$$  
(5)

where $\Upsilon = (\tau(v_i, v_j))_{n \times n}$ is defined in Eq. (4), $H^{(2)}$ are the weight parameters to be trained, and $Z = (z_{ic})_{n \times K}$ the final community membership of nodes (where $z_{ic}$ denotes the probability that node $v_i$ belongs to community $c_i$).

3 The Method

After a brief overview of our new method, we will introduce an encoder based on MRFasGCN to derive node community memberships and a dual decoder for reconstructing links and attributes based on communities. We then discuss a regularization to enhance community detection locally.

3.1 Overview

The new GCN-based approach for Unsupervised Community Detection, short-handed as GUCD, adopts Autoencoder as its overall architecture and includes three main parts (Fig. 1). In the first part (the green box to the left of Fig. 1), we adopted the three convolutional layers of MRFasGCN as the encoder of GUCD, where the first two layers were to learn a deep representation of the attribute network and the third layer was to model and derive node community membership using both the deep representation and network information. We then designed a dual decoder as the second part of GUCD (the red box to the right of Fig. 1), using the derived communities to separately reconstruct network topology and node attributes.
We first reconstructed the network topology by requiring the nodes within the same community to maintain the same link pattern (with also considering the heterogeneity of node degrees) connecting to the rest of the network, which is suitable for generating the coupling relationships between nodes. We then generated node contents using topic modelling, i.e. we assumed that the contents of nodes in the same community share similar distributions of attribute words used. We formulated the dual decoder to fully utilize (structural and content) data from diverse sources, making the decoding process suitable for unsupervised community detection. In the third part (in purple box in the middle of Fig. 1), we added a regularization of local enhancement to the latent communities, i.e. we first constructed an aggregated graph combining the information of both topology and attributes, and then made each pair of connected nodes in this graph have similar community distributions. The model was trained as a whole using the Adam optimizer [Kingma and Ba, 2015].

### 3.2 The Shared Encoder

We processed the network topology and node attributes together using the same encoder so as to extract hidden network characteristics, particularly latent community structures to be identified. To be specific, we adopted the three convolutional layers of MRFasGCN [Jin et al., 2019] as the encoder (the green box of Fig. 1). We used the first two layers to derive a general embedding and then derived an initial node community labels $X^{(2)}$ (defined in Eq. (1)) by using softmax on the embedding, which is not community-specific. We used this initial solution $X^{(2)}$ to define unary potentials in MRF, and then defined pairwise potentials of MRF to model communities $Z$ (defined in Eq. (5)) hidden in network topology and attributes, leading to the third convolutional layer. Through these three layers, the derived node community labels $Z$ not only utilize the deep representation but also are smoothly community-oriented. The node to community assignments in $Z$ will be learned when the whole model with encoder and decoder is trained together; the high quality of the reconstructed network will require an accurate community structure at the end of the encoder.

### 3.3 The Dual Decoder

The dual decoder is the core part of the new method. It consists of two decoders, one for reconstructing network topology and the other for reconstructing node attributes.

#### The Decoder for Reconstructing Network Topology

This novel decoder attempts to reconstruct network topology based on the node community membership derived in the latent space. It not only makes the decoder community-oriented, but also achieves unsupervised learning for community detection. The idea is inspired by the block model for blocks, groups, or communities in networks. That is, if the community which a node $v_i$ belongs to is denoted as $c_i$ ($c_i = 1, 2, \ldots, K$), we can then define a $K \times K$ block matrix $\Theta$ such that each element $\theta_{rs}$ in the matrix is the possibility of having an edge between any two nodes $v_i$ and $v_j$ (with $c_i = r$ and $c_j = s$). In this case, the nodes in each of the $K$ communities (with label $r$) have the same link pattern, i.e., the nodes in the same community share the same link probability with any node $v_j$ in the network, i.e. $\theta_{rc_j}$. This idea naturally describes the coupling relationship between nodes in the network since it is defined to generate pairwise rather than individuals based on community structure. The model is further improved by considering the heterogeneity of node degrees, i.e. the nodes with higher degrees should be more likely to be connected. Therefore, the model can be revised such that the possibility that nodes $v_i$ and $v_j$ are connected is $d_i d_j \theta_{rc_j}$, where $d_i$ is the degree of $v_i$. This mechanism is in concordance with the degree-corrected stochastic block model (DCSBM) [Karrer and Newman, 2011] even though the formulation is different.

Based on the above model, the expected number of links between nodes $v_i$ and $v_j$, which respectively belong to communities $c_i$ and $c_j$, can be written as

$$\pi_{ij}^{c_i c_j} = \text{sigmoid} \left( z_{ic_i} z_{jc_j} d_i d_j \theta_{c_j c_j} \right)$$

where $z_{ic_i}$ is the probability that $v_i$ belongs to community $c_i$, which is from the encoder, defined in Eq. (5). Considering
all communities \( \{ (c_i, e_j) | c_i, e_j \in L \} \), we can then define the expected number of links between nodes \( v_i \) and \( v_j \) as:

\[
\pi_{ij} = \text{sigmoid} \left( \sum_{c_i=1}^{K} \sum_{c_j=1}^{K} z_{ic_i} z_{jc_j} d_i d_j \theta_{c_i c_j} \right)
\]  

(7)

By considering both the block modelling and heterogeneity degree of nodes, the model can well describe the coupling relationship among nodes in the network with community structures. The above model can be formulated as a layer of a neural network to serve as an attributed decoder. Using the Decoder for Reconstructing Attributes

The Decoder for Reconstructing Attributes

The above generative process can be formulated as a layer of neural network to serve as an attributed decoder. Using the topology of node community memberships from the encoder; and the adjacency matrix \( \tilde{A} \) generated from the model. This difference can be defined by using the cross-entropy loss as:

\[
L_{\text{topo}} = - \sum_{i,j=1}^{n} \left[ a_{ij} \ln \pi_{ij} + (1 - a_{ij}) \ln (1 - \pi_{ij}) \right]
\]  

(9)

3.4 Local Enhancement

Different from the main model (e.g., the MRF part) which describes communities via global information, we further utilize local information to enhance communities from a local view. The rationale is that two nodes should have similar node community membership if they are close to each other, topologically and/or semantically, in the attribute network.

We implemented this local enhancement scheme by introducing pairwise constraints on nodes and a graph regularization term to the objective function. We first built an aggregated graph based on network topology and node attributes. To measure the proximity of a pair of nodes, we introduced a measurement method to combine their topological and attribute similarities. We computed the set of the local neighbors for each node \( v_i \), i.e.,

\[
\Gamma_i^{\text{topo}} = \Gamma_i^{\text{topo}} \cup \Gamma_i^{\text{attr}}
\]  

(15)

where \( \Gamma_i^{\text{topo}} \) is the set of neighbors directly adjacent to \( v_i \), and \( \Gamma_i^{\text{attr}} \) the set of top-\( b_{\text{attr}} \) most attribute-similar neighbors of \( v_i \), where the TF-IDF cosine similarity is used to calculate the attribute similarity of nodes. That is, let \( \vec{x}_i \) be the attribute vector of node \( v_i \), then for attribute unit \( x_{it} \), its TF-IDF value in a term vector \( \vec{x}_i \) is:

\[
x_{it} = tf - idf \left( x_{it}, \vec{x}_i \right)
\]  

\[
= \sqrt{tf \left( x_{it}, \vec{x}_i \right) \cdot \log \left( 1 + \frac{|V|}{\sum_{j=1}^{|V|} tf \left( x_{it}, \vec{x}_j \right)} \right)}
\]  

(16)

The cosine similarity of attributes between nodes \( v_i \) and \( v_j \) is then defined as:

\[
\text{cosine} \left( \vec{x}_i, \vec{x}_j \right) = \vec{x}_i \cdot \vec{x}_j / \left( \left\| \vec{x}_i \right\| \cdot \left\| \vec{x}_j \right\| \right)
\]  

(17)

Thereafter, we added weights to the aggregated graph. We calculated the proximity between a node and its direct neighbors in this new graph using:

\[
S = \lambda \text{zero-one} \left( \text{S}^{\text{topo}} \right) + (1 - \lambda) \text{zero-one} \left( \text{S}^{\text{attr}} \right)
\]  

(18)
plays the role of normalization defined as:

\[ S_{ij}^{\text{topo}} = \cosine(\hat{a}_i, \hat{a}_j) \]

and

\[ S_{ij}^{\text{attr}} = \cosine(\hat{x}_i, \hat{x}_j) \]

are the topological and attribute similarities between node \( v_i \) and its direct neighbor \( v_j \in \Gamma_i \). \( \lambda \) is a tradeoff parameter, and \( \text{zero-one}(\cdot) \) plays the role of normalization defined as:

\[ \text{zero-one}(\hat{y}) = (y_i - \min(\hat{y})) / (\max(\hat{y}) - \min(\hat{y})) \]

Given the similarity matrix \( S \) of the aggregated graph, we defined the pairwise constraint as:

\[ L_{\text{reg}} = \sum_{i,j} s_{ij} \| z_i - z_j \|_2^2 = 2 \text{tr} \left( Z^T \Psi Z \right) \]

where \( \Psi = F - S \) and \( F \) is a diagonal matrix with \( f_{ii} = \sum_j s_{ij} \). We then incorporated the pairwise constraint as a graph regularization in the objective function to enhance community detection locally.

Finally, the objective of the model is to minimize the following loss function:

\[ L = \gamma L_{\text{topo}} + (1 - \gamma) L_{\text{attr}} + \eta L_{\text{reg}} \]

where \( \gamma \) and \( \eta \) are the parameters that control the trade-off between different parts of the loss. We used the back-propagation (BP) algorithm and Adam optimizer to train the model. At convergence, the algorithm produces the community label for each node by:

\[ \hat{c}_i = \arg \max_{c_i \in L} z_{ic_i} \]

### 4 Experiments

We compared our approach with eight state-of-the-art methods on nine widely-used benchmark datasets. We also analyze the features of our method to appreciate its effectiveness.

#### 4.1 Experiment Setup

To carry out an accurate comparison of the nine methods, we used nine public datasets with known communities (Table 1). As the networks used have ground-truth communities, we adopted two widely used metrics, i.e. accuracy (AC) [Liu et al., 2012] and normalized mutual information (NMI) [Danon et al., 2005], for performance evaluation. We applied Adam optimizer in our GUCD method using TensorFlow.

#### 4.2 Comparison with the Existing Methods

We first evaluated our GUCD against eight (unsupervised) community detection methods. Depending on what network information they use, these existing methods can be grouped into three types. The first type uses only network topology, which includes DCSBM [Karrer and Newman, 2011] and EdMot-SC [Li et al., 2019]. The second uses only node attributes, including LDA [Blei et al., 2003]. The third uses topology and attributes together, including Block-LDA [Balasubramanyan and Cohen, 2011], PCLDC [Yang et al., 2009], SCI [Wang et al., 2016], MISAGA [He and Chan, 2018] and TLSC [Zhang et al., 2018].

GUCD is the best on 7 and 6 out of 9 datasets in terms of AC (Table 2) and NMI (Table 3). On the remaining networks where GUCD does not perform the best, it is still competitive with the best baselines. These results demonstrate the superiority of our new approach over the existing methods.

#### 4.3 Deep Analysis of GUCD

Similar to most existing deep learning methods, GUCD has multiple components affecting its performance. Moreover, MRFasGCN is a major component of GUCD and can be applied in different ways. Here we report the results from different combinations of different major components of GUCD.

### Effects of Individual Components

We compared GUCD with six variations of GUCD. We first considered two unsupervised variants of MRFasGCN, a major component of GUCD: 1) the weight parameters \( H^{(0)} \), \( H^{(1)} \), and \( H^{(2)} \) in MRFasGCN is set to 1 without training (as the most general way), namely MasG-U1, and 2) the weight parameters is set randomly without training (as suggested as an unsupervised usage) namely MasG-U2. We included in our comparison with another unsupervised variant of MRFasGCN by adding the inner product of node community membership for reconstructing links as the decoder, namely MasG-IN. (Note that this is the most common way used in unsupervised GCN models for network embedding, a different albeit similar problem.) We also tested three variants of GUCD. The first two reconstruct links and attributes separately, named as GUCD-1 and 2. The third is GUCD without the local enhancement, called GUCD-3.

The experimental results on nine benchmark problems revealed that GUCD performed the best on 7 and 7 out of
the 9 networks in terms of AC (Table 4) and NMI (Table 5). Specifically, GUCD is on average 17.45% (and 24.49%) and 23.99% (and 29.18%) more accurate than MasG-U1 and MasG-U2 in AC (and NMI). By some extra experiments we also observed that to achieve similar performances with GUCD, MRFasGCN needs almost 30%, 30%, 30%, 30%, 20%, 0.2%, 0.8%, 2% and 0.05% supervised information for the nine datasets respectively in training. This validates not only the effectiveness of our unsupervised framework using Autoencoder, but also the soundness of the community-oriented dual decoder, i.e. we use different while the most suitable mechanisms to reconstruct links and attributes respectively for community detection. Besides, GUCD is on average 17.3% (and 22.55%) more accurate than MasG-IN in AC (and NMI). This further validates the soundness of the new decoding mechanisms (i.e. we reconstruct links by modelling that nodes in the same community share the same link pattern, and reconstruct attributes based on topic modelling) over the existing methods for reconstructing links alone via the inner product operation. In addition, GUCD is on average 15.99% (and 22.4%), 2.35% (and 1.06%), and 1.71% (and 3.4%) more accurate than GUCD-1, GUCD-2, and GUCD-3 in terms of AC (and NMI). This demonstrates that the topological and attributed decoders both are effective. The strategy for local enhancement also helps mitigate overfitting caused by the sparsity of networked data.

### Qualitative Analysis

To further validate the effectiveness of our new decoding mechanism over the one that most commonly used in network embedding, we further compared GUCD and MasG-IN by showing the results of AC and NMI as a function of the number of iterations. Consider the results on Twitter as an example. As shown in Fig. 2, the GUCD results match more closely to the ground truth than MasG-IN, and the result of the former is more stable than that of the latter with more training cycles, suggesting that the new decoding mechanisms are suitable for unsupervised community detection.

### 5 Related Work

The current work of GCN focuses primarily on node classification [Kipf and Welling, 2017], which has been adopted for semi-supervised community detection [Li et al., 2018; Jin et al., 2019]. However, community detection is, in essence, an unsupervised problem since little training data are available for most applications. It is imperative to develop novel GCN methods for unsupervised community finding.

A related line of work is network embedding using GCN. For example, the ARGA method [Pan et al., 2018] adopts an adversarially regularized graph Autoencoder, which uses the same encoder and decoder as GAE [Kipf and Welling, 2016] while introduces an adversarial module to force the node representation to follow a suitable prior distribution. The ARGA method has also been extended to use graph convolution, instead of inner product, as the decoder to reconstruct links; the resulting method can reconstruct links better than reconstructing both links and attributes [Pan et al., 2019]. For unsupervised learning, most of the existing methods focus primarily on reconstructing network structures, as their performance degrades when being used to reconstruct links and attributes together. These existing methods seem to be suitable for representation learning, e.g., embedding, but not adequate for unsupervised community detection, a more difficult learning task. This has been experimentally shown by the comparison between GUCD and its variant MasG-IN (using inner product to reconstruct links), as shown in Fig. 2 and Tables 4 & 5.

### 6 Conclusion and Discussion

This is the first approach extending GCN effectively to unsupervised community detection. Using the Autoencoder framework, we focused on community identification rather than network embedding in the encoder; and used the most suitable mechanisms to reconstruct links and attributes separately. We also introduced a local enhancement to ameliorate the issue of overfitting brought by the sparsity of networked data. We carried out experiments and demonstrated the superiority of the new approach. We further showed the effectiveness of the novel decoding mechanism for generating links and attributes against the most commonly used methods for reconstructing links alone. Even though the new approach was designed for community detection, the underlying idea may be readily extended to GCN-based network embedding as discussed in Related Work.

### Acknowledgments

This work was supported by the Natural Science Foundation of China (61832014, 61876128, 61772361, U1736103).
References


