Information Sciences 608 (2022) 1464–1479

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Semi-supervised overlapping community detection in attributed graph with graph convolutional autoencoder

Chaobo He [a,b], Yulong Zheng [a], Junwei Cheng [a,b], Yong Tang [a,b], Guohua Chen [a,b], Hai Liu [a,*]

[a] School of Computer Science, South China Normal University, Guangzhou 510631, China
[b] Pazhou Lab, Guangzhou 510335, China

ARTICLE INFO

Article history:
Received 16 December 2021
Received in revised form 11 April 2022
Accepted 5 July 2022
Available online 09 July 2022

Keywords:
Overlapping community detection
Semi-supervised learning
Graph convolutional autoencoder
Graph neural networks
Attributed graph

ABSTRACT

Community detection in attributed graph is of great application value and many related methods have been continually presented. However, existing methods for community detection in attributed graph still cannot well solve three key problems simultaneously: link information and attribute information fusion, prior information integration and overlapping community detection. Aiming at these problems, in this paper we devise a semi-supervised overlapping community detection method named SSGCAE which is based on graph neural networks. This method is composed of three modules: graph convolutional autoencoder (GCAE), semi-supervision and modularity maximization, which are respectively utilized to fuse link information and attribute information, integrate prior information and detect overlapping communities. We treat GCAE as the backbone framework and train it by using the unified loss from these three modules. Through this way, these three modules are jointly correlated via the community membership representation, which is very beneficial to improve the overall performance. SSGCAE is comprehensively evaluated on synthetic and real attributed graphs, and experiment results show that it is very effective and outperforms state-of-the-art baseline approaches.

1. Introduction

Many real-world complex networks (e.g., social networks, co-authorship networks and biological information networks) are often naturally modeled as graphs, which makes graph data become ubiquitous and meanwhile promotes the emergence of various topics of graph data mining. Community detection, also known as graph clustering, is one of long-standing popular research topics. It aims to discover node clusters with high cohesiveness, that nodes in the same cluster should connect to each other more densely than those in different clusters [8]. Community detection not only can help to uncover the structure features of the graph, but also has important practical significance and application value, such as fraud detection [26], terrorist organization identification [27] and social recommendation [1].

In the past few decades, many community detection methods have been proposed and demonstrate different degrees of performance. However, by analyzing some comprehensive literature on community detection methods (e.g., [7,15]), we can find that most of existing methods only utilize the single link information in graph, which makes them hard to identify communities accurately when encountering graphs containing incomplete and even noisy link information. To overcome this drawback, some researchers noticed the graph associated with node attributes or content information (i.e., attributed graph)

* Corresponding author.
E-mail addresses: hechaobo@foxmail.com (C. He), 1457367033@qq.com (Y. Zheng), jung@m.scnu.edu.cn (J. Cheng), ytang@m.scnu.edu.cn (Y. Tang), chengh3@qq.com (G. Chen), liuhai@m.scnu.edu.cn (H. Liu).

https://doi.org/10.1016/j.ins.2022.07.036
0020-0255/

and creatively proposed community detection methods that can integrate both link information and attribute information, such as SA-Cluster [4] and CODICIL [28]. Due to the compensation effect of attribute information on link information, this type of methods often can achieve better performance than methods using link information alone. Moreover, they can provide a semantic interpretation to the resultant communities by extracting strongly correlated attributes [4,13].

Nowadays, the attributed graph is becoming more and more common. Meanwhile, community detection in this type of graph has also drawn increasing attention and various methods have appeared. In [5], Chunaev conducted a deep comparative analysis to many representative community detection methods in attributed graph, from which we realize that there are three significant problems as follows that have not been well solved yet.

- Fusing link information and attribute information. The ideal solution to fuse these two types of information is with a seamless and automatic way. However, most existing methods need to set the hyper parameters manually to balance the contributions of link information and attribute information. This requires a lot of tuning cost to obtain the optimal parameters on different graph datasets. Moreover, link information and attribute information cannot compensate each other automatically.
- Integrating prior information (e.g., known node-community labels) that some graph nodes may have. Many community detection methods using only link information, such as NMF_KL [43] and PSSNMF [24], have proved that utilizing available prior information via semi-supervised learning models can dramatically increase the performance. Unfortunately, existing community detection methods in attributed graph pay relatively little attention to this and are almost unsupervised without considering any prior information.
- Detecting overlapping communities. Most existing methods mainly focus on identifying non-overlapping communities. However, overlapping communities are also common in real attributed graphs. For example, nodes in the attributed graph constructed from online social networks are often assigned with multiple group labels, which denote the corresponding user belongs to multiple communities.

As another emerging and highly relevant topic, Graph Neural Networks (GNNs) have recently received unprecedented attention in the field of graph data mining, and its various variants show superior performance in many graph-related tasks like graph embedding, node classification, link prediction, etc., all of which are comprehensively surveyed in [39,49,48]. As the important features, GNNs can provide a seamless and automatical framework to fuse link information and attribute information, and also are very good at semi-supervised learning on graph. These inspire us to exploit GNNs to solve the aforementioned problems facing by existing community detection approaches in attributed graph.

In this paper, we develop a GNN-based method for semi-supervised overlapping community detection in attributed graph. For convenience, we call this method SSGCAE for short here after. The main works are summarized as follows:

- We devise a simple yet effective graph convolutional autoencoder (GCAE). Through this GNN variant, link information and attribute information can be fused seamlessly and automatically to learn the accurate community membership representation.
- To directly obtain semi-supervised overlapping community detection results with an end-to-end manner, we devise a goal-oriented optimization framework to train GCAE by jointly optimizing graph reconstruction loss, modularity maximization loss and semi-supervision loss.
- We conduct extensive experiments on several synthetic and real attributed graphs. The results show that our proposed method SSGCAE outperforms state-of-the-art community detection methods in attributed graph.

The rest of this paper is organized as follows. In Section 2 we briefly review the related work. The details on the proposed method SSGCAE and the experiment results are given in Section 3 and Section 4, respectively. We finally conclude this paper in Section 5.

2. Related work

In this section, we overview some works that are most relevant to ours from two aspects: community detection in attributed graph and GNNs.

2.1. Community detection in attributed graph

More recently, some methods for community detection in attributed graph have been proposed. According to how to fuse link information and attribute information. These methods can be roughly divided into two categories. One is based on the unified similarity or distance function for these two types of information. Representative methods include SA-Cluster [4], CODICIL [28] and ANCA [6]. The other type of methods is based on the specific learning model, such as probabilistic model based method GBAGC [41], canonical correlation analysis (CCA) [44] based method CTTA [12] and Nonnegative Matrix Factorization (NMF) based methods SCI [35] and CFOND [10].

Although existing methods possess different levels of information fusion ability, most of them need to expensively tune the hyper parameters that balance the contributions of link information and attribute information. For example, CODICIL [28] needs to manually set a weight parameter to construct the similarity fusion matrix used for subsequent community detection, and SCI [35] also has a regularization parameter applied to control the contribution of link information matrix factorization term. This way that cannot fuse information seamlessly may make these methods inefficient and even ineffective. Furthermore, when link information or attribute information has noises, these two types of information are hard to compensate each other automatically.

For overlapping communities detection and prior information integration, some methods for community detection in attributed graph have paid attention to these two problems, but most of them solve them separately. For example, methods NOCD [30], MOEA-SA$_{OA}$ [33] and CommunityGAN [16] only focus on detecting overlapping communities, while SCHAIN [23], PSSNMTF [17] and MRFasGCN [18] only focus on improving the performance of community detection by incorporating prior information. Unlike these methods, our proposed method SSGCAE simultaneously takes overlapping communities detection and prior information integration into account.

2.2. GNNs

GNNs are the products of applying deep learning techniques to graph representation learning. By using multilayer neural networks, it can learn more informative and discriminative node representation than traditional methods [42], which makes it popularly used in the domain of graph data mining. Recently, many GNN variants have been continuously presented, which mainly include four types: Graph Recurrent Neural Networks (GRNNs) [11,29], Graph Attention Networks (GATs) [34,47], Graph Convolutional Networks (GCNs) [20,46] and Graph AutoEncoders (GAEs) [21,31]. Although these GNNs have different model architectures, they basically have two common operations to learn node representation: propagating information along the link structure and aggregating information from neighboring nodes. In particular, attribute information is often treated as the initial source of node representation, and hence GNNs actually provide a framework to fuse link information and attribute information automatically and seamlessly without any balancing parameters. Lots of existing work has also proved that this way can help GNNs further improve node representation, which is more beneficial to the downstream tasks [39,49,48].

In the past few years, GNNs have received increasing attention in many graph-related tasks, including community detection focused on in this paper. Both [32,19] survey existing representative GNN-based methods for community detection in attributed graph. Although these methods all achieve a certain degree of performance improvement via the powerful representation learning ability of GNNs, they also completely ignore or only partially solve the problems of overlapping communities detection and prior information integration. For example, both DAEGC [36] based on GAT and SDCN [2] based on GCN do not focus on these two problems, and NOCD [30] and MRFasGCN [18] just respectively address the problems of overlapping communities detection and prior information integration.

To the best of our knowledge, there is still a lack of methods that can triply solve the problems of fusing link information and attribute information, integrating prior information and detecting overlapping communities. In this paper, with the aid of GNNs, our proposed method SSGCAE not only fuse link information and attribute information in an automatic and seamless manner, but also can integrate prior information and detect overlapping communities simultaneously. Compared with existing methods for community detection in attributed graph, SSGCAE can be expected to perform more effectively and versatilely.

3. Methodology

In this section, we first introduce the notations and preliminaries used in this paper, and then describe the details of our proposed method SSGCAE.

3.1. Notations and preliminaries

Throughout this paper, we denote matrices by bold uppercase letters. For a given matrix $\mathbf{Z}$, its i-th row vector, (i,j)-th element, trace, transpose and Frobenius norm are denoted as $\mathbf{Z}_i, \mathbf{Z}_{ij}, tr(\mathbf{Z}), \mathbf{Z}^T$ and $||\mathbf{Z}||_F$, respectively. All frequently used notations in this paper and their descriptions are summarized in Table 1.

The preliminaries are introduced as follows, including related definitions and the problem statement.

Definition 1 (Attributed graph): Without loss of generality, here we consider undirected and unweighted attributed graphs. A given attributed graph is denoted as $\mathcal{G} = (V, E, \mathbf{A}, \mathbf{X})$, where $V = \{v_1, v_2, \ldots, v_n\}, E = \{e_{ij} | v_i \in V \wedge v_j \in V\}, \mathbf{A} = [\mathbf{A}_{ij}]^{n\times n}$ is the adjacency matrix and $\mathbf{X} = [\mathbf{X}_{ip}]^{n\times s}$ is the node-attribute matrix. If $e_{ij} \in E, \mathbf{A}_{ij} = 1$, and $\mathbf{A}_{ij} = 0$ if $e_{ij} \notin E$. For the attributes set $\mathcal{A} = \{a_1, a_2, \ldots, a_s\}$, if v_i has attribute $a_p, \mathbf{X}_{ip} = 1$, and $\mathbf{X}_{ip} = 0$ otherwise. $\mathbf{A}$ and $\mathbf{X}$ are used to represent link information and attribute information of $\mathcal{G}$, respectively.

Definition 2 (Community): A community C_i is a subgraph of $\mathcal{G}$, where nodes link to each other more densely than those outside. Supposing that $\mathcal{G}$ is partitioned into k communities, we denote the communities set as $\mathcal{C} = \{C_i | C_i \neq \emptyset, 1 \leqslant i \leqslant k\}$. We consider the possibility of communities overlapping, and hence the intersection of C_i and C_j may not be empty for $\forall i \neq j$. For ease of presentation, we also treat the community index i as the corresponding community label.

Table 1
Notations and their descriptions.

Notations	Descriptions
$\mathscr{G}$	A given attributed graph
V	Nodes set
v_i	The i-th node
E	Edges set
e_{ij}	The edge between v_i and v_j
n	The number of nodes
m	The number of edges
k	The number of communities
$\mathscr{C}$	Communities set
C_i	The i-th community
$\mathbf{X}$	The node-attribute matrix
$\mathscr{A}$	Attributes set
s	The size of attributes set
$\mathbf{A}$	The adjacency matrix
$\mathbf{Y}$	The prior node-community labels matrix
$\mathbf{H}$	The final node representation matrix, also the community membership representation matrix
$\mathbf{I}$	The identity matrix
$\mathbf{W}^{(l)}$	The weight matrix of the l-th layer
$\mathbf{H}^{(l)}$	The node representation matrix of the l-th layer
d_l	The dimension of node representation of the l-th layer
L	The number of neural networks layers
$\sigma(\cdot)$	The activation function
$\mathbb{R}_+$	The nonnegative real number set

Definition 3 (Prior information): Here we consider the known node-community labels information as the prior information, which means that some nodes have ground-truth community labels (i.e., $\mathscr{G}$ is partially labeled). These information can be encoded as a prior node-community labels matrix $\mathbf{Y} = [\mathbf{Y}_{iq}]^{n\times k}$. If v_i belongs to community $C_q, \mathbf{Y}_{iq} = 1$, and $\mathbf{Y}_{iq} = 0$ otherwise.

Based on the definitions above, the community detection problem that this paper wants to solve is formulated as follow:

Problem statement: Given an attributed graph $\mathscr{G}$ and its available link information $\mathbf{A}$, attribute information $\mathbf{X}$ and prior information $\mathbf{Y}$, dividing its nodes into k communities $C_1, C_2, \ldots, C_k$ by means of the specially developed GNN, where communities are allowed to be overlapped.

3.2. Overview

The main idea of SSGCAE is presented in Fig. 1. As shown in this figure, SSGCAE comprises three main modules: graph convolutional autoencoder (GCAE), modularity maximization and semi-supervision.

We first develop GCAE composed of an encoder used to learn a latent representation $\mathbf{H}$ of $\mathscr{G}$, and a decoder used to reconstruct the link structure of $\mathscr{G}$. Through GCAE, link information and attribute information can be fused seamlessly and automatically. Then, in order to enable $\mathbf{H}$ to also indicate the community membership, modularity maximization module is introduced to drive GCAE to learn a refined $\mathbf{H}$, which is capable of reflecting the discriminative community structure. Further, semi-supervision module imposes constraints from prior information on $\mathbf{H}$ to make it become more accurate. Finally, to obtain the optimal $\mathbf{H}$, GCAE is trained via the unified loss function $\mathscr{L}$ composed of graph reconstruction loss $\mathscr{L}_{\mathbf{REC}}$, modularity maximization loss $\mathscr{L}_{\mathbf{MM}}$ and semi-supervision loss $\mathscr{L}_{\mathbf{SS}}$. In what follows, we will introduce more details on SSGCAE.

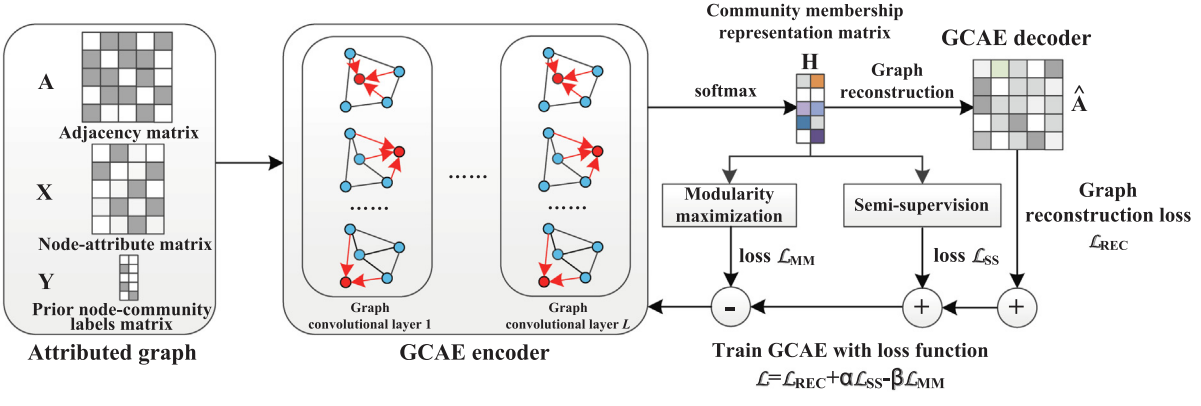


Fig. 1. The architecture of SSGCAE. Note here that **A**, **X** and **Y** are toy examples of adjacency matrix, node-attribute matrix and prior node-community labels matrix, respectively.

3.3. GCAE module

GCAE is the basis of SSGCAE, and consists of an encoder and a decoder. For the encoder, we adopt a multiple-layer Graph Convolutional Network (GCN) with the following layer-wise propagation rule:

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

where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix with self loops, and $\tilde{\mathbf{D}}$ is the diagonal degree matrix of $\tilde{\mathbf{A}}$ with $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$. Here, we select $\mathrm{ReLU}(x) = \max(0, x)$ as the activation function. Especially, for the last layer L of GCN, we use $\mathrm{softmax}(x_i) = \frac{1}{\mathscr{Z}}\exp(x_i)$ with $\mathscr{Z} = \sum_i \exp(x_i)$ as the activation function to obtain the final representation $\mathbf{H}$. Namely, we have

$$\mathbf{H} = \mathrm{softmax}(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(L-1)}\mathbf{W}^{(L)}). \tag{2}$$

Note that we set the input of the first layer of GCN to be $\mathbf{X}$, i.e., $\mathbf{H}^{(0)} = \mathbf{X}$. Therefore, through the propagation and aggregation mechanism shown in Eq. (1), link information $\mathbf{A}$ and attribute information $\mathbf{X}$ of $\mathscr{G}$ can be fused seamlessly and automatically. Since $\mathbf{H}$ can well preserve the proximity between nodes, for the decoder of GCAE, it is natural to use $\mathbf{H}$ to reconstruct the link structure of $\mathscr{G}$. Specifically, we can reconstruct an adjacency matrix $\hat{\mathbf{A}}$ to approximate to the original adjacency matrix $\mathbf{A}$. Every element $\hat{\mathbf{A}}_{ij}$ in $\hat{\mathbf{A}}$ is treated as the probability of whether v_i and v_j are connected or disconnected (denoted as 1 or 0), which is respectively computed as follows:

$$p(\hat{\mathbf{A}}_{ij} = 1) = \mathrm{sigmoid}(\mathbf{H}_i\mathbf{H}_j^T), \tag{3}$$

$$p(\hat{\mathbf{A}}_{ij} = 0) = 1 - \mathrm{sigmoid}(\mathbf{H}_i\mathbf{H}_j^T), \tag{4}$$

where $\mathrm{sigmoid}(x) = \frac{1}{1+e^{-x}}$. We apply the following binary cross entropy function to measure the reconstruction loss between $\hat{\mathbf{A}}$ and $\mathbf{A}$, i.e., graph reconstruction loss $\mathscr{L}_{\mathrm{REC}}$:

$$\mathscr{L}_{\mathrm{REC}} = -\frac{1}{n^2}\sum_{i,j}(\mathbf{A}_{ij}\log p(\hat{\mathbf{A}}_{ij} = 1) + (1 - \mathbf{A}_{ij})\log p(\hat{\mathbf{A}}_{ij} = 0)). \tag{5}$$

By minimizing $\mathscr{L}_{\mathrm{REC}}$, GCAE can be trained to achieve the goal that making $\hat{\mathbf{A}}$ closer to $\mathbf{A}$.

3.4. Modularity maximization module

For GCAE encoder, if we set the dimension of its final layer to the number of communities k (i.e., $\mathbf{H} \in \mathbb{R}_+^{n\times k}$), $\mathbf{H}$ can also be treated as the community membership representation matrix, where every row vector represents the community membership strength distribution of the corresponding node. However, if we utilize GCAE alone, $\mathbf{H}$ only can provide a coarse representation of community structure. After all, such GCAE is not community detection oriented. In view of this, we incorporate the following modularity maximization model to further refine $\mathbf{H}$ for obtaining more discriminative community membership representation.

Modularity maximization proposed by Newman [25] has been widely used to model the community structure. Formally, for a network with $\mathbf{H} \in \mathbb{R}_+^{n\times k}$ as the community membership representation matrix, its modularity Q is defined as follows:

$$Q = \frac{1}{4m}\sum_{i,j}^{n}(\mathbf{A}_{ij} - \frac{d_i d_j}{2m})(\mathbf{H}_i\mathbf{H}_j^T), \tag{6}$$

where d_i is the degree of v_i. By defining the modularity matrix $\mathbf{B} = [\mathbf{B}_{ij}]^{n\times n}$, whose element $\mathbf{B}_{ij} = \mathbf{A}_{ij} - \frac{d_i d_j}{2m}$, we can derive the trace form of modularity Q via the following steps:

$$\begin{aligned} Q &= \tfrac{1}{4m}\sum_{i,j}^{n}(\mathbf{B}_{ij}\sum_{p}^{k}\mathbf{H}_{ip}\mathbf{H}_{jp}) \\ &= \tfrac{1}{4m}\sum_{p}^{k}\sum_{i,j}^{n}\mathbf{B}_{ij}\mathbf{H}_{ip}\mathbf{H}_{jp} \\ &= \tfrac{1}{4m}\sum_{p}^{k}[\mathbf{H}^T\mathbf{B}\mathbf{H}]_{pp} \\ &= \tfrac{1}{4m}tr(\mathbf{H}^T\mathbf{B}\mathbf{H}). \end{aligned} \tag{7}$$

After suppressing the constant $\frac{1}{4m}$ which has no effect on the maximum of the modularity, we can obtain the loss of modularity maximization $\mathscr{L}_{\mathrm{MM}}$ as:

$$\mathscr{L}_{\mathrm{MM}} = tr(\mathbf{H}^T\mathbf{B}\mathbf{H}). \tag{8}$$

Note that the original modularity maximization model is for detecting nonoverlapping communities, and requires that: in each row of $\mathbf{H}$, only one element should be 1 and the rest should be 0. Namely, it has the constraint that $tr(\mathbf{H}^T\mathbf{H}) = n$. How-

ever, to facilitate uncovering overlapping communities, here we relax this constraint and allow elements in $\mathbf{H}$ to take values in the range of $[0, 1]$, which is achieved via softmax function in GCAE. In practice, this modularity maximization model is also very effective for detecting overlapping communities, which will be verified in our experiments.

3.5. Semi-supervision module

This module aims to incorporate known node-community labels information $\mathbf{Y}$ to further enhance the results of community detection. When $\mathbf{Y}$ is available, it means that $\mathcal{G}$ is partially labeled. Therefore, we can apply the widely used cross-entropy loss as the semi-supervision loss $\mathcal{L}_{\text{SS}}$:

$$\mathcal{L}_{\text{SS}} = -\sum_{i\in\mathcal{Y}}\sum_{j=1}^{k}\mathbf{Y}_{ij}\ln\mathbf{H}_{ij}, \tag{9}$$

where $\mathcal{Y}$ is the set of node indices that have labels. By minimizing $\mathcal{L}_{\text{SS}}$, the available prior information can be expected to further boost the performance of community detection.

3.6. The unified loss function

As analyzed above, using GCAE alone only learns a coarse community membership representation matrix $\mathbf{H}$, because GCAE is not community detection oriented. However, if we integrate GCAE, modularity maximization and semi-supervision modules into a whole, we can expect to obtain a more accurate $\mathbf{H}$. To this end, we combine the GCAE loss $\mathcal{L}_{\text{REC}}$ together with the modularity maximization loss $\mathcal{L}_{\text{MM}}$ and the semi-supervision loss $\mathcal{L}_{\text{SS}}$ to construct the unified loss function:

$$\mathcal{L} = \mathcal{L}_{\text{REC}} + \alpha\mathcal{L}_{\text{SS}} - \beta\mathcal{L}_{\text{MM}}, \tag{10}$$

where α and β are positive hyper-parameters for adjusting the contributions of corresponding modules, the joint effects of which will be analyzed in our experiments. By training GCAE with $\mathcal{L}$, GCAE, modularity maximization and semi-supervision modules can promote each other. Furthermore, GCAE becomes community detection oriented, and provides an end-to-end solution to the problem of semi-supervised community detection in attributed graph.

3.7. Overlapping community detection algorithm and complexity analysis

After being trained until the maximum iterations, GCAE will obtain a stable $\mathbf{H}$. To uncover overlapping communities, we can set a threshold φ to determine the community memberships of nodes. Specifically, if $\mathbf{H}_{ip} \geqslant \varphi, v_i$ will be assigned into community C_p. This strategy is simple yet effective, and is also widely adopted by soft clustering based methods for community detection [40]. Actually, overlapping community detection itself is a soft clustering problem. To specify φ properly, we take the same method proposed by Yang and Leskovec [45], and calculate φ as:

$$\varphi = \sqrt{-\log(1-\frac{2m}{n(n-1)})}. \tag{11}$$

This threshold strategy works well in practice. Following the idea above, we devise the whole SSGCAE overlapping community detection algorithm presented in Algorithm 1.

Algorithm 1: SSGCAE overlapping community detection [9]

Input: Attributed graph $\mathcal{G} = (V, E, \mathbf{A}, \mathbf{X})$, Prior information $\mathbf{Y}$,
Number of communities k, Maximum iterations $MaxIter$;
Output: Communities set $\mathcal{C} = \{C_1, C_2, ..., C_k\}$;

```
Initialize W^(l) in GCAE with Xavier initialization method [9];
for t = 1...MaxIter do
    Generate H with GCAE encoder;
    Feed H to GCAE decoder to construct Â;
    Calculate L_REC, L_MM and L_SS, respectively;
    Calculate the unified loss function L via Eq. (10);
    Back propagation and update W^(l) in GCAE;
for v_i ∈ V do
    for ∀H_ip ∈ H_i do
        if H_ip ≥ sqrt(−log(1 − 2m/(n(n−1)))) then C_p = C_p ⋃ v_i;
return C;
```

As shown in Algorithm 1, its main computational cost is largely concentrated in the iterative part from line 2 to line 7. Because GCN can be efficiently implemented using sparse matrix multiplication and its time complexity is linear with the number of edges m, the time complexity of GCAE encoder can be computed as $O(msd_1d_2\dots d_L)$. As for reconstructing $\hat{\mathbf{A}}$, and calculating $\mathscr{L}_{\text{REC}}, \mathscr{L}_{\text{MM}}$ and $\mathscr{L}_{\text{SS}}$, the dominant time complexity is $O(n^2k)$. In total, the time complexity of SSGCAE is $O(msd_1d_2\dots d_L + n^2k)$.

4. Experimental study

To validate the effectiveness of SSGCAE, in this section we first conduct comparison analysis with five baselines on several synthetic and real attributed graphs to demonstrate the superiority of SSGCAE, and then deeply analyze its parameter sensitivity and running efficiency. We implement all methods using Python 3.7 and use Deep Graph Library (DGL) [37] to develop GNNs. All experiments are conducted on a PC with 64-bit Windows 10 system, 3.5 GHz Intel Core i9-11900 K CPU and 64 GB RAM.

4.1. Baselines

As discussed in Section 1 and Section 2, existing community detection methods in attributed graph all have different degrees of fusion capability of link information and attribute information, and most of them completely ignore or only partially solve the other two problems: detecting overlapping communities and integrating prior information. However, we find that some of them can easily be transformed to detect overlapping communities by introducing the same threshold strategy as SSGCAE. Therefore, to facilitate the comparison, from these methods we select two types of state-of-the-art methods as baselines: unsupervised and semi-supervised, according to the principle that whether to integrate prior information or not.

The unsupervised methods include:

- SCI [35]. SCI adopts joint NMF framework to fuse link information and attribute information. It learns a soft membership distribution matrix over communities, which can be exploited to uncover overlapping communities.
- NMFjGO [14]. NMFjGO is the combination of joint NMF and graph optimization. It can dynamically fuse link information and attribute information by introducing dynamic embedding learning procedure, and also can detect overlapping communities.
- NOCD [30]. NOCD is based on GCN framework, which also can seamlessly and automatically fuse link information and attribute information. Without labeling partial nodes in advance, NOCD utilizes graph reconstruction loss produced by Bernoulli-Poisson probabilistic model for overlapping community [45] to train GCN, which makes it unsupervised and easy to detect overlapping communities directly.

Recently, semi-supervised methods for community detection in attributed graph are very scarce. Considering the potential ability to detect overlapping communities, we select the following two semi-supervised methods:

- WSCDSM [38]. WSCDSM provides a unified NMF framework to integrate link information, attribute information and prior information, and the overlapping community structure can be identified from its learnable community membership matrix.
- PSSNMTF [17]. Unlike most of NMF-based methods, PSSNMTF devises a semi-supervised NMF tri-factorization model with node popularity, which helps it not only can better fuse link information, attribute information and prior information, but also can discover overlapping communities.

These comparative methods above all have some parameters needed to be preseted. For our proposed method SSGCAE, we set $\alpha = 10^{-2}$ and $\beta = 10^{-3}$, the network configuration of GCN in its GCAE module is: $256 - k$ and the learning rate is set to $5e^{-3}$. For baseline methods, the parameters of them are set to their suggested values and we also further carefully tune parameters to get optimal performance.

4.2. Evaluation metrics

We mainly evaluate the performance of comparative methods in terms of accuracy. Specifically, aiming to the detection accuracy of overlapping communities, we introduce two widely used evaluation metrics: Overlapping Normalized Mutual Information (ONMI) [3] and average F1-score (F1 for short) [45]. Given the set of ground-truth communities $\mathscr{C}\prime$ and the set of detected communities $\mathscr{C}$, ONMI is defined as follows:

$$\text{ONMI} = 1 - \frac{1}{2}\left(\sum_{i}^{|\mathscr{C}|}\frac{H(C_i|\mathscr{C}\prime)}{|\mathscr{C}|H(C_i)} + \sum_{i}^{|\mathscr{C}\prime|}\frac{H(C_i\prime|\mathscr{C})}{|\mathscr{C}\prime|H(C_i\prime)}\right) \tag{12}$$

where $H(C_i)$ denotes the entropy of community C_i:

$$H(C_i) = -\frac{|C_i|}{n}\log\frac{|C_i|}{n}, \tag{13}$$

and $H(C_i|\mathscr{C}\prime)$ denotes the conditional entropy of C_i on $\mathscr{C}\prime$:

$$H(C_i|\mathscr{C}\prime) = \min_{q\in\{1,2,\ldots,|\mathscr{C}\prime|\}} H(C_i|C_{q\prime}). \tag{14}$$

F1 is defined to be the average of the F1-score of between $\forall C_i \in \mathscr{C}$ and $\forall C_i\prime \in \mathscr{C}\prime$ as follows:

$$\mathrm{F1} = \frac{1}{2}\left(\frac{1}{|\mathscr{C}\prime|}\sum_{i}^{|\mathscr{C}\prime|}\mathrm{F1}(C_i\prime, C_{g(i)}) + \frac{1}{|\mathscr{C}|}\sum_{i}^{|\mathscr{C}|}\mathrm{F1}(C_i, C_{g\prime(i)\prime})\right), \tag{15}$$

where the best matching $g(i)$ and $g\prime(i)$ are respectively defined as $g(i) = \mathrm{argmax}_j \mathrm{F1}(C_i\prime, C_j)$ and $g\prime(i) = \mathrm{argmax}_j \mathrm{F1}(C_i, C_j\prime)$. $\mathrm{F1}(C_i\prime, C_j)$ is the harmonic mean of Precision and Recall, which are both frequently-used evaluation metrics for information retrieval.

To evaluate whether the genuine communities can be identified or not, in our experiments the number of detected communities k is set to the ground-truth. Namely, we have $k = |\mathscr{C}| = |\mathscr{C}\prime|$. According to definitions, ONMI and F1 are both in the range of $[0, 1]$. Larger values of them indicate better ability of overlapping community detection. For a fair comparison, we run every method 10 times and the average results are reported here.

4.3. Experimental analysis

In this part, we first conduct comparative experiments on synthetic and real attributed graphs, and demonstrate the superiority of SSGCAE from two aspects: prior information integration ability and information fusion ability. Next, we perform an ablation study of SSGCAE to confirm that modularity maximization module plays the key role in uncovering accurate overlapping community structure. Finally, we respectively carry out parameter sensitivity analysis and running efficiency analysis of SSGCAE.

4.3.1. Experiments on synthetic attributed graphs

(1) Datasets. We utilize the well-known LFR model [22] to generate synthetic graphs with ground-truth overlapping communities. By respectively varying some key parameters, and fixing other parameters of LFR model, we generate 2 groups of synthetic graphs: SG1 and SG2. The detail parameter settings of every group are shown in Table 2. As the post-process step, we adopt the following strategy to generate the corresponding attribute information for every LFR synthetic graph. Firstly, we set the size of attributes set s to $k \times h$, where h denotes the uniform size of community attributes. This means that every node has a $k \times h$ dimensional attribute vector. Next, for the given v_i with ground-truth community index set Q, its $((q-1)\times h+1)$-th to $(q \times h)$-th attributes are set to 1 with probability P_{in}, where $q \in Q$. Finally, the rest attributes of v_i are set to 1 with probability P_{out} ($P_{\text{in}} + P_{\text{out}} = 1$). This attribute generation strategy fully considers the homophily of community members in real attributed graphs. Namely, nodes in the same community should share more common attributes than those outside. In order to present this strategy more clearly, we give an illustrative example shown in Fig. 2.

For all LFR synthetic graphs in our experiments, we set $h = 5, P_{\text{in}} = 0.8$ and $P_{\text{out}} = 0.2$. We call these generated synthetic graphs with attribute information as synthetic attributed graphs. Every synthetic attributed graph has ground-truth communities, from which prior information can be extracted. Specifically, according to the preassigned ratio, we randomly label some nodes with ground-truth community indexes. Then we can construct the known node-community labels matrix $\mathbf{Y}$ based on these prior information. Note that $\mathbf{Y}$ is easily converted to must-link constraints used in WSCDSM and PSSNMTF methods. For example, if $\mathbf{Y}_{ij} = 1$, then we can generate the corresponding must-link constraint that v_i and v_j should be in the same community.

(2) Prior information integration ability comparison. We select group SG1 which actually has only one synthetic attributed graph, and test the performance of every method by varying the ratio of prior information from 0% to 10% with a step size of 2%. The results are shown in Fig. 3, from which we can find that:

- SSGCAE can be improved greatly with a small proportion of prior information. When the ratio of prior information is 0% (i.e., semi-supervised methods also do not utilize any prior information), all semi-supervised methods are inferior to the best unsupervised method: SCI. However, when the ratio of prior information is 2%, SSGCAE has significant improvement: ONMI and F1 respectively improve by 0.76 and 0.84, and meanwhile performs better than SCI and other semi-supervised methods.
- SSGCAE outperforms other methods by a large margin when utilizing prior information. Specifically, when the ratio of prior information varies from 2% to 10%, comparing with SCI, NMFjGO, NOCD, WSCDSM and PSSNMTF, the average ONMI score of SSGCAE is respectively 0.18, 0.54, 0.21, 0.25 and 0.36 higher, and its average F1 score is respectively 0.41, 0.68, 0.64, 0.69 and 0.73 higher. It should be pointed out that the performance of SSGCAE can be continuously improved with the increase of prior information, but other semi-supervised methods WSCDSM and PSSNMTF are not always the case.

Table 2
Parameter settings of LFR synthetic graphs (The step size of *mu* in SG2 is 0.1).

Parameter	Explanation	SG1	SG2
n	Number of nodes	3000	5000
mu	Mixing paramter	0.3	[0.3, 0.7]
on	Number of overlapping nodes	$\frac{n}{10}$	
om	Number of overlapping memberships	3	
d_{avg}	Average degree	20	
d_{max}	Maximum degree	50	
λ_1	Exponent for node degree distribution	2	
λ_2	Exponent for community size distribution	1	
C_{min}	Minimum community size	$\frac{n}{50}$	
C_{max}	Maximum community size	$\frac{n}{10}$	

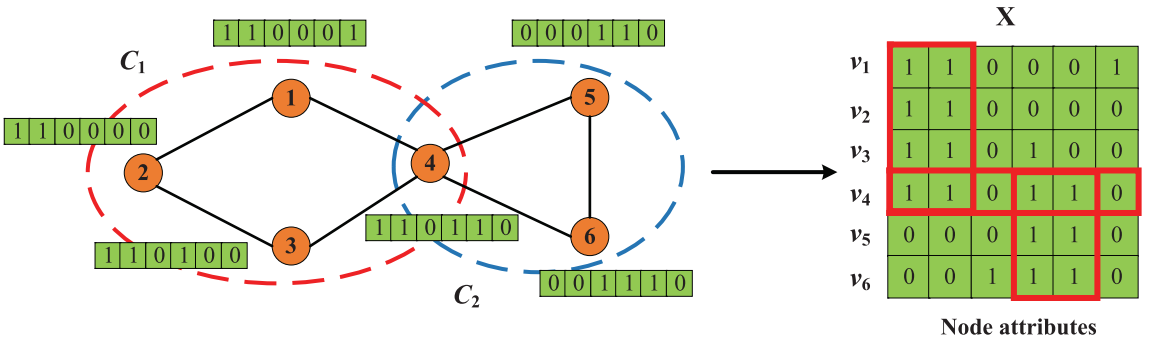


Fig. 2. Illustration of the attribute generation strategy to preserve homophily of community members. Here, $h = 3, P_{in} = 0.8, P_{out} = 0.2$. $C_1 = \{v_1, v_2, v_3, v_4\}$ and $C_2 = \{v_4, v_5, v_6\}$. Obviously, v_1, v_2 and v_3 share more common attributes (highlighted with red box) than v_5 and v_6, and vice versa. As the overlapping node, v_4 is similar to all other nodes..

(3) Information fusion ability comparison. Good information fusion ability can make link information and attribute information compensate each other automatically to improve the final performance as much as possible, especially when they have noises. To validate this, we first select SG2 synthetic attributed graphs with 2% prior information, and vary *mu* from 0.3 to 0.7 with a step size of 0.1. A larger *mu* means more noisy link information is added and detecting communities becomes more difficult. The results on these graphs are illustrated in Fig. 4. As we can see, although the performance of each method begins to decrease with the increase of *mu*, SSGCAE shows more stable and better performance than other methods. In particular, when $mu = 0.7$ that means link information no longer manifests distinct community structure, SSGCAE still achieves satisfactory performance: ONMI = 0.66 and F1 = 0.75. On the contrary, the performance of other methods is extremely low and almost negligible.

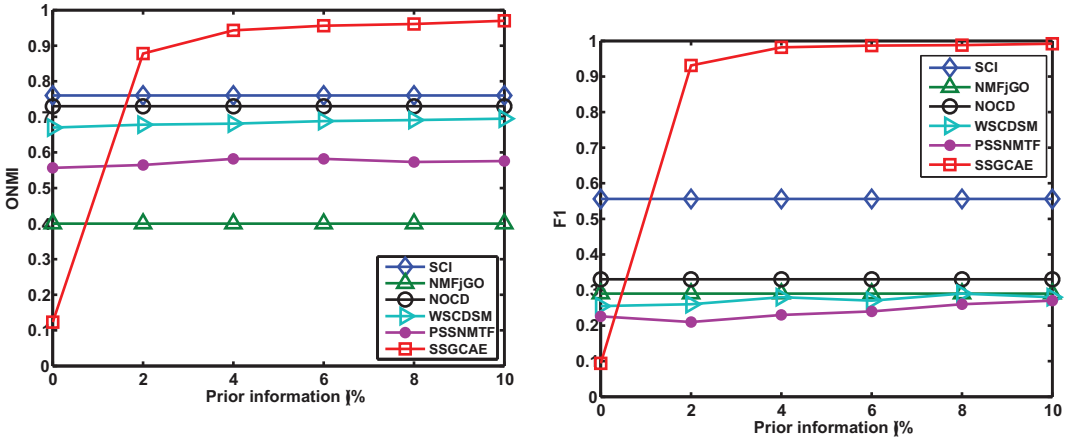


Fig. 3. Performance comparison on SG1 synthetic attributed graphs with different ratios of prior information.

We then select an attributed graph from SG2 with $mu = 0.3$ and 2% prior information, and select its $P_{\text{mis}}\%$ nodes to randomly exchange attribute vectors with each other, which can simulate adding noisy attribute information into the graph. We vary P_{mis} from 20 to 60 with a step size of 10 and the corresponding results are presented in Fig. 5. We can find that SSGCAE consistently maintains stable and satisfactory performance, but most other methods deteriorate very severely, especially semi-supervised methods WSCDSM and PSSNMTF. Although SCI also performs stably, but its performance is far inferior to this of SSGCAE.

4.3.2. Experiments on real attributed graphs

(1) Datasets. In [30], Oleksandr et al. constructed benchmark real datasets used for evaluating the performance of overlapping community detection in attributed graph, from which we select four real attributed graphs with ground-truth overlapping communities. Statistics of these graphs are provided in Table 3.

(2) Prior information integration ability comparison. Just like the experiment types conducted on synthetic attributed graphs, we first test the performance of each method on every real attributed graph, accompanying by varying the ratio of

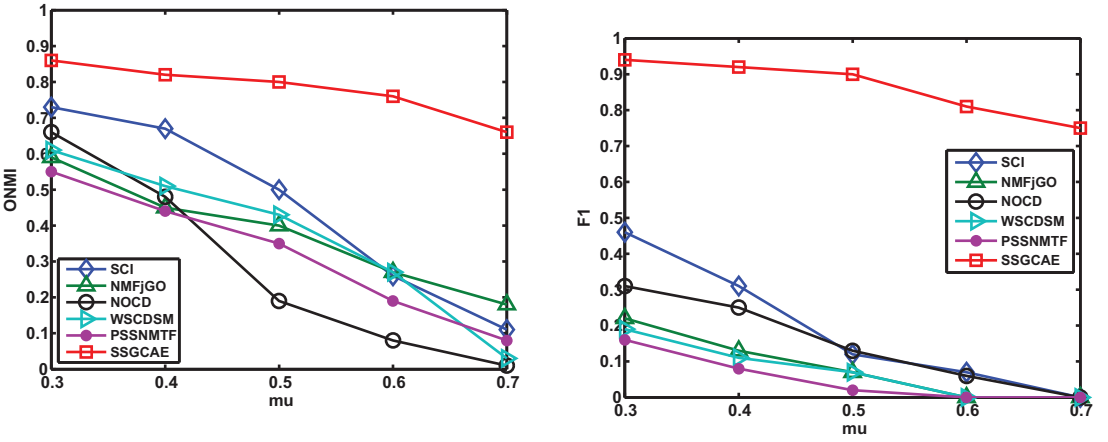


Fig. 4. Performance comparison on SG2 synthetic attributed graphs with different *mu*.

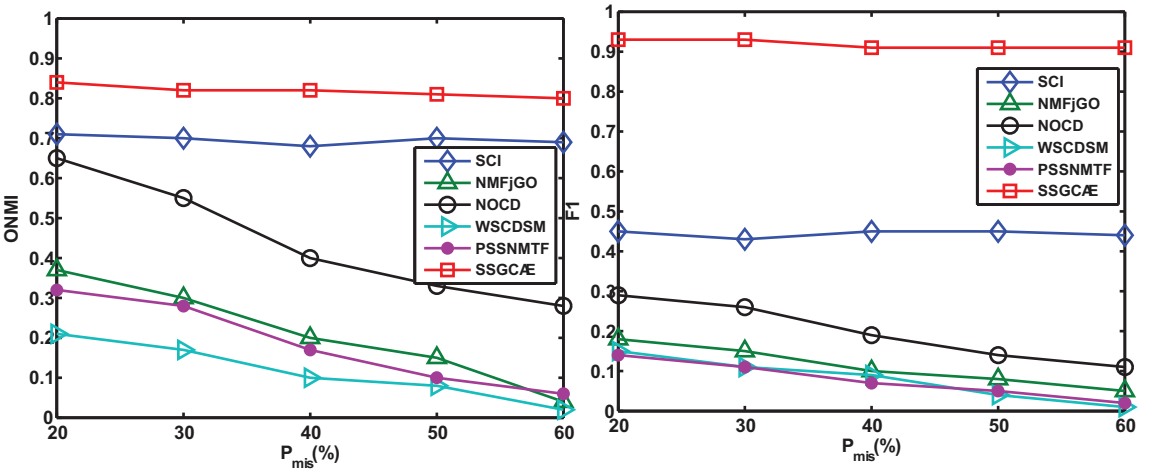
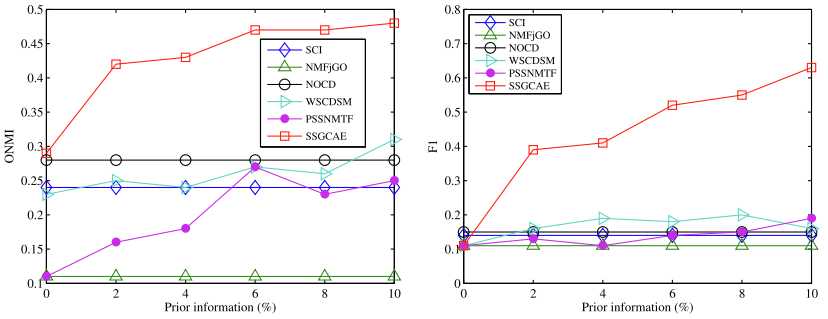


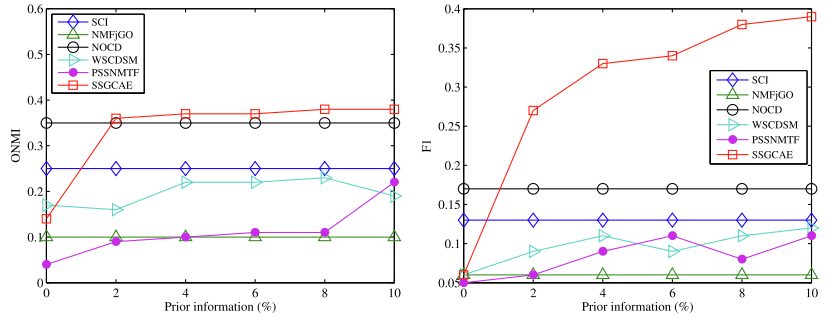
Fig. 5. Performance comparison on SG2 synthetic attributed graphs with different P_{mis}.

Table 3
Statistics of real attributed graphs.

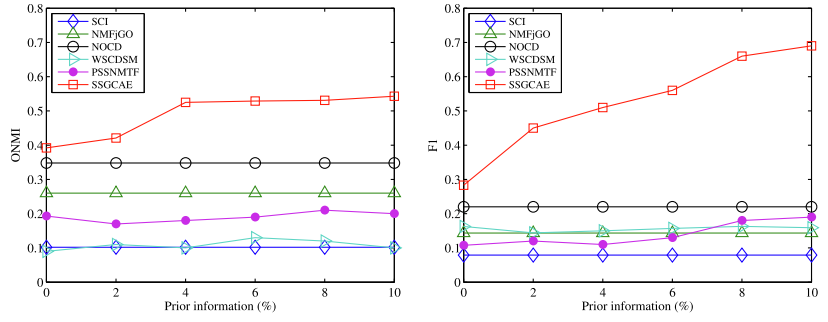
Dataset	Graph type	n	m	s	k
Facebook 1684	Social	792	28,048	15	17
Facebook 1912	Social	755	60,050	29	46
Computer science	Co-authorship	21,957	193,500	7,800	18
Engineering	Co-authorship	14,927	98,610	4,800	16



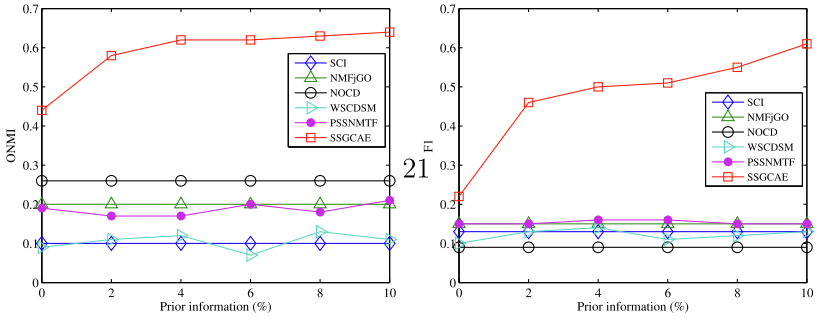
(a) Facebook 1684



(b) Facebook 1912



(c) Computer science



(d) Engineering

Fig. 6. Performance comparison in terms of ONMI and F1 on real attributed graphs with different ratios of prior information.

prior information from 0% to 10% with a step size of 2%. The results are shown in Fig. 6. As we can see, on every dataset, the more prior information, the more superior the performance of SSGCAE.

(3) Information fusion ability comparison. We take Computer science attributed graph with 2% prior information as an example, and first select its P_{mis}% nodes to randomly exchange attribute vectors with each other (P_{mis}% also varies from 20 to 60 with a step size of 10), and then run every method on this graph. The results are presented in Fig. 7, from which we observe the same phenomena as those happened on synthetic attributed graphs: the higher the P_{mis}, the worse the performance of most baselines, but SSGCAE still can achieve more stable and satisfactory performance. To further show the advantage of SSGCAE, in Table 4 we list the results of all methods on every real attributed graph when $P_{\text{mis}} = 60$. We can clearly see that all method except SSGCAE perform quite poorly.

The aforementioned comparison results on synthetic and real attributed graphs show that SSGCAE can better integrate prior information, and fuse link information and attribute information to boost the performance of overlapping community detection. This largely benefits from the more powerful semi-supervised learning ability and information fusion ability of GCN used in GCAE module. Besides, most baselines are all based on linear models (e.g., NMF adopted by SCI, WSCDSM and PSSNMTF) which are hard to process various nonlinear features of attributed graph like GCN. Although NOCD also utilizes GCN, it is unsupervised so that prior information cannot be used to further boost its performance.

4.3.3. Ablation experiments

As discussed in Section 3.4, modularity maximization module is used to refine **H** to uncover community structure. To verify this effect, we remove this module from SSGCAE and call this version as SSGCAE-Mod, and then compare it with SSGCAE on four real attributed graphs, all of which has 2% prior information. The results are shown in Fig. 8, from which we can see that SSGCAE outperforms SSGCAE-Mod significantly on each dataset. For example, compared to SSGCAE-Mod, on Computer science the ONMI and F1 scores of SSGCAE respectively improve by 55.6% and 95.7%.

These results above indicate that SSGCAE identifies more accurate community structure than SSGCAE-Mod. Considering that only SSGCAE introduces modularity maximization module, we conclude that this module can lead to better performance. Without the guidance of this module, SSGCAE-Mod only can output coarse and indiscriminative community membership representation.

4.3.4. Parameter sensitivity analysis

In the objective function Eq. (10) of SSGCAE, parameters α and β respectively control the contributions of the semi-supervision module and modularity maximization module. To explore their effects on the performance of SSGCAE, we conduct empirical analysis on Facebook 1684 by simultaneously varying them in the range of $\{10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$ and fixing the ratio of prior information at 2%. The joint effects of α and β are shown in Fig. 9. As can be seen, SSGCAE is not very sensitive to α and β. As long as α and β do not take too large values simultaneously, SSGCAE is able to achieve satisfactory performance. Similar results can be observed on other graphs, we omit them here to avoid repetition. In all our experiments, we set $\alpha = 10^{-2}$ and $\beta = 10^{-3}$.

4.3.5. Running efficiency analysis

We first select the relatively larger real attributed graph Computer science with 2% prior information to perform convergence analysis of SSGCAE. The results are presented in Fig. 10, from which we can see that SSGCAE converges rapidly. Only

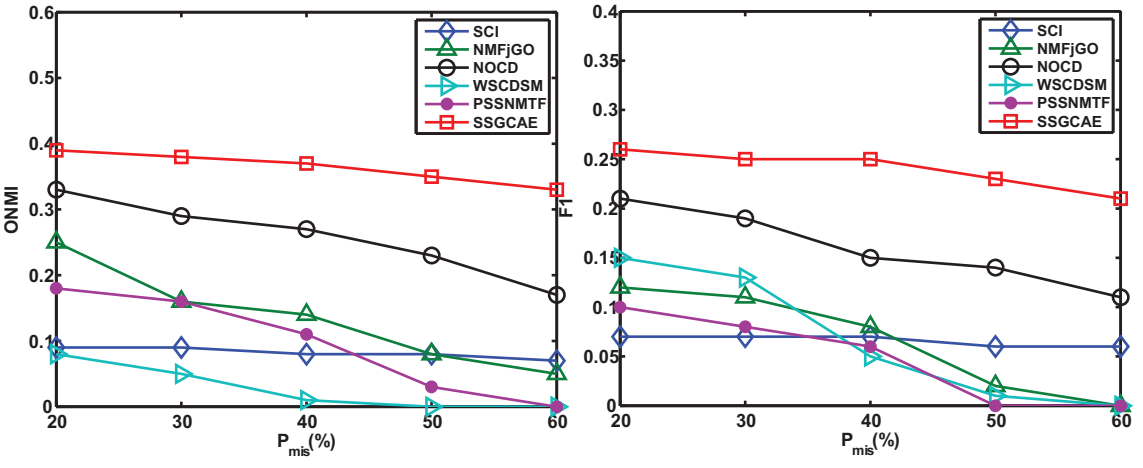


Fig. 7. Performance comparison on Computer science with different P_{mis}.

Table 4
Performance comparison on real attributed graphs when $P_{mis} = 60$. (Bold numbers denote the best results)

Methods	Facebook 1684		Facebook 1912		Computer science		Engineering	
	ONMI	F1	ONMI	F1	ONMI	F1	ONMI	F1
SCI	0.09	0.08	0.06	0.04	0.07	0.06	0.02	0.01
NMFjGO	0.02	0.01	0.04	0.01	0.05	0.00	0.00	0.00
NOCD	0.13	0.12	0.15	0.09	0.17	0.11	0.12	0.08
WSCDSM	0.02	0.01	0.03	0.01	0.00	0.00	0.00	0.00
PSSNMTF	0.05	0.03	0.04	0.00	0.00	0.00	0.00	0.00
SSGCAE	**0.36**	**0.27**	**0.31**	**0.22**	**0.33**	**0.21**	**0.43**	**0.39**

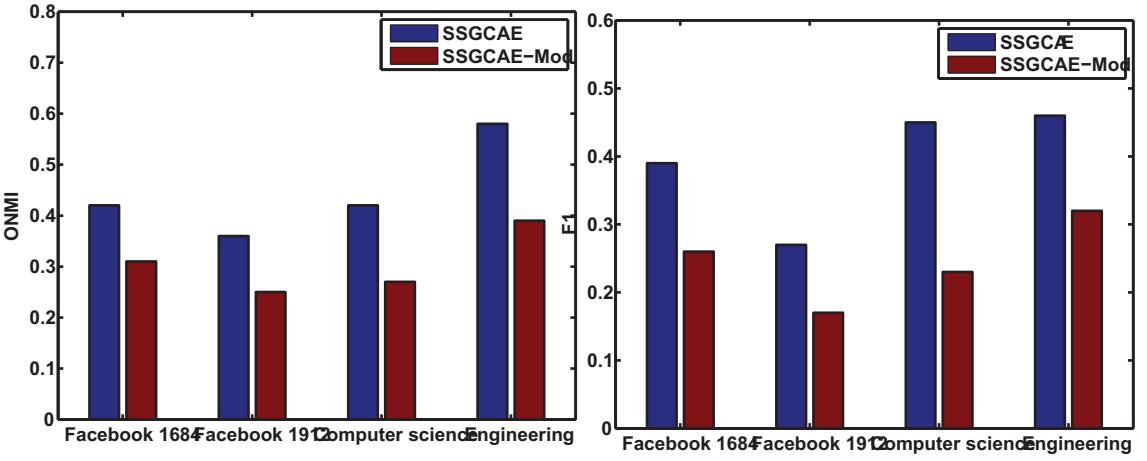


Fig. 8. Performance comparison between SSGCAE and SSGCAE-Mod on real attributed graphs.

within a few iterations (about 30 iterations on Computer science), the loss of objective function (Fig. 10 (a)) becomes stable, and the performance (Fig. 10 (b)) reaches the best.

We then compare the running time of different methods on all real attributed graphs with 2% prior information and the results are shown in Table 5. As we can see, on Facebook 1684 and Facebook 1912, although SSGCAE is a little inferior to all unsupervised methods, it runs faster than semi-supervised methods WSCDSM and PSSNMTF. On Computer science and Engineering, SSGCAE is the second best that is only inferior to NOCD. Especially, comparing with WSCDSM and PSSNMTF, SSGCAE has obvious advantages. For example, on Computer science, SSGCAE is respectively about 1.6 and 22 times faster than WSCDSM and PSSNMTF. Both WSCDSM and PSSNMTF have too many dense matrix multiplication operations, which are very time consuming. Although the calculation of modularity in SSGCAE also refers to the same operations and results in the quadratic time complexity in theory, they can be implemented via high efficient sparse-dense tensor multiplication of

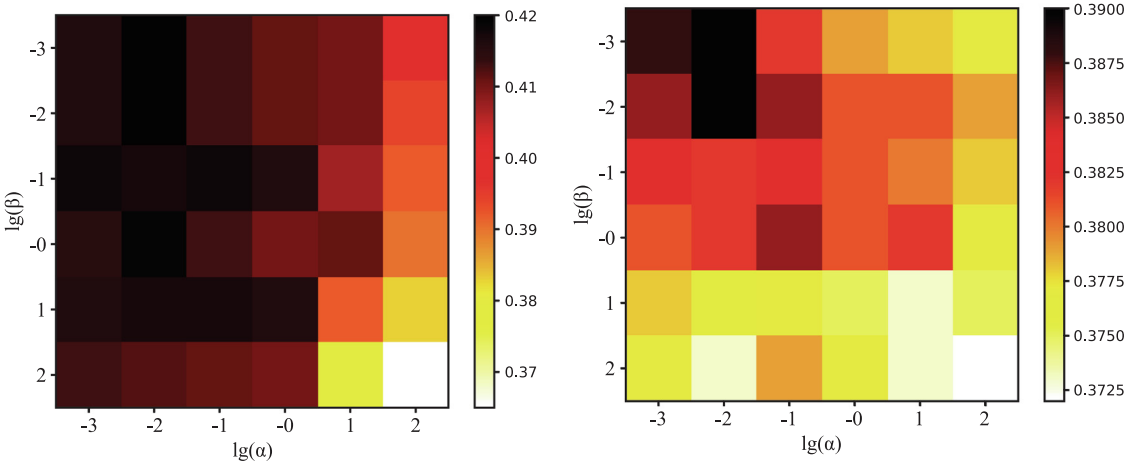


Fig. 9. Joint effects of α and β on Facebook 1684.

Python due to the fact that the involved matrix **H** is very sparse: only having very few non-zero elements representing the community membership strength.

Based on the analysis above, in practice SSGCAE has the potential to deal with larger attributed graphs. To verify this, we run it on a synthetic attributed graph with 100,000 nodes and 1,954,016 (nearly 2 million) edges, which is generated using the same parameter settings (except n) and attribute generation strategy as SG1 in Table 2. We find that it also takes about 30 iterations to converge (Fig. 11 (a)) and the convergence time is about 7.3 h. Specially, we monitor the memory cost at every iteration and the result is shown in Fig. 11 (b). As can be seen, our 64 GB memory is enough, and when SSGCAE begins to converge, its memory cost also begins to become stable. It is worth pointing out that other baselines except NOCD all run out of memory.

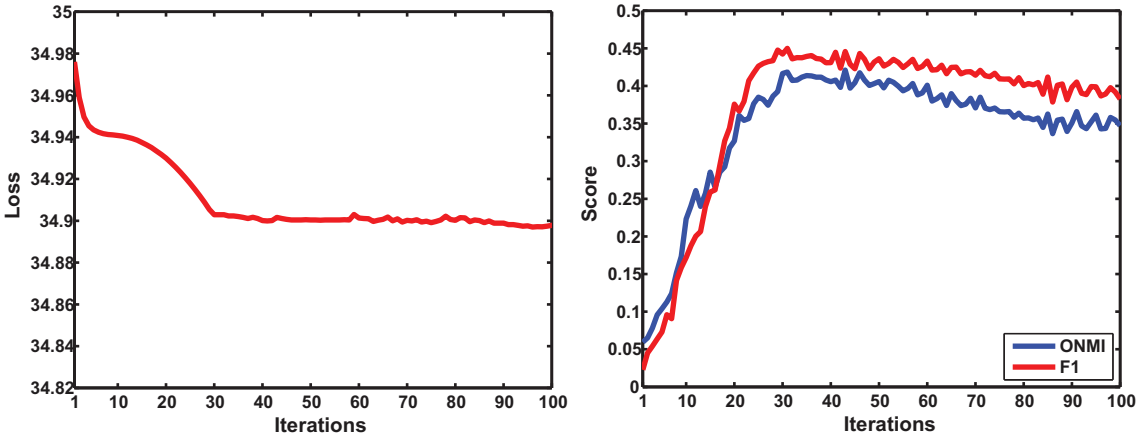


Fig. 10. Convergence analysis on Computer science.

Table 5
Running time (Seconds) comparison on real attributed graphs with 2% prior information.

Methods	Facebook 1684	Facebook 1912	Computer science	Engineering
SCI	1	4	3,013	1,109
NMFjGO	3	5	5,002	2,122
NOCD	4	7	268	118
WSCDSM	85	287	6,183	2,103
PSSNMTF	13	45	54,480	28,683
SSGCAE	10	37	2,344	871

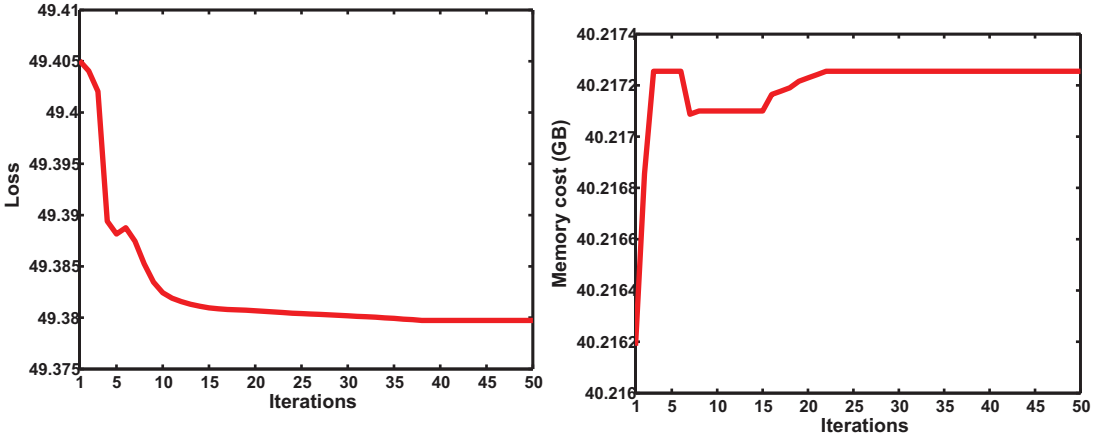


Fig. 11. Efficiency analysis on a large-scale synthetic attributed graph with 100,000 nodes and 1,954,016 edges.

5. Conclusion

To detect more accurate overlapping communities in attributed graphs, in this paper we propose a semi-supervised learning method SSGCAE based on the framework of GCAE, which is a variant of GNNs. With the aid of GCAE, link information and attribute information can be better fused, and meanwhile prior information can be integrated more effectively. Extensive experiments are conducted on both synthetic and real attributed graphs, and the results comprehensively verify the effectiveness and efficiency of SSGCAE. In summary, SSGCAE provides a simple yet effective solution to semi-supervised overlapping community detection in attributed graph.

In the future, we plan to further extend SSGCAE from two meaningful directions. One is extending it to detect overlapping semantic communities, which can be described with the most relevant attributes or topics. The other is extending it to detect overlapping communities in dynamic attributed graph by introducing temporal GNNs. Moreover, we also will seek to further improve its efficiency by devising more efficient GNNs and community detection models.

CRediT authorship contribution statement

Chaobo He: Writing - original draft, Conceptualization, Methodology. **Yulong Zheng:** Software, Visualization. **Junwei Cheng:** Software, Visualization. **Yong Tang:** Supervision. **Guohua Chen:** Conceptualization. **Hai Liu:** Supervision, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62077045, Grant U1811263 and Grant 61772211, in part by the Humanity and Social Science Youth Foundation of Ministry of Education of China under Grant 19YJCZH049, and in part by the Natural Science Foundation of Guangdong Province of China under Grant 2019A1515011292.

References

[1] A. Alsini, A. Datta, D.Q. Huynh, On utilizing communities detected from social networks in hashtag recommendation, IEEE Transactions on Computational Social Systems 7 (4) (2020) 971–982.
[2] D.Y. Bo, X. Wang, C. Shi, M.Q. Zhu, E. Lu, P. Cui, Structural deep clustering network, in: Proceedings of the 29th International Conference on World Wide Web (WWW), ACM, 2020, pp. 1400–1410.
[3] T. Chakraborty, A. Dalmia, A. Mukherjee, N. Ganguly, Metrics for community analysis: a survey, ACM Computing Surveys 50 (4) (2017) 54.
[4] H. Cheng, Y. Zhou, J.X. Yu, Clustering large attributed graphs: a balance between structural and attribute similarities, ACM Transactions on Knowledge Discovery from Data 5 (2) (2011) 12.
[5] P. Chunaev, Community detection in node-attributed social networks: a survey, Computer Science Review 37 (2020) 100286.
[6] I. Falih, N. Grozavu, R. Kanawati, Y. Bennani, Community detection in attributed network, in: Proceedings of the 27th International Conference on World Wide Web (WWW), ACM, 2018, pp. 1299–1306.
[7] S. Fortunato, D. Hric, Community detection in networks: a user guide, Physics Reports 659 (2016) 1–44.
[8] M. Girvan, M.E.J. Newman, Community structure in social and biological networks, Proceedings of the National Academy of Sciences 99 (12) (2002) 7821–7826.
[9] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AIS), 2010, pp. 249–256.
[10] T. Guo, S. Pan, X. Zhu, C. Zhang, CFOND: consensus factorization for co-clustering networked data, IEEE Transactions Knowledge Data Engineering 31 (4) (2019) 706–719.
[11] E. Hajiramezanali, A. Hasanzadeh, K. Narayanan, N. Duffield, M.Y. Zhou, X.N. Qian, Variational graph recurrent neural networks, in: Proceedings of the 33rd Conference in Neural Information Processing Systems (NeurIPS), 2019, pp. 1–11.
[12] Y. He, C. Wang, C.J. Jiang, Discovering canonical correlations between topical and topological information in document networks, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM), 2015, pp. 1281–1290.
[13] X. Huang, H. Cheng, J.X. Yu, Dense community detection in multi-valued attributed networks, Information Sciences 314 (2015) 77–99.
[14] Z.H. Huang, X.X. Zhong, Q. Wang, M.G. Gong, X.K. Ma, Detecting community in attributed networks by dynamically exploring node attributes and topological structure, Knowledge-Based Systems 196 (2020) 105760.
[15] M.A. Javed, M.S. Younis, S. Latif, J. Qadir, A. Baig, Community detection in networks: a multidisciplinary review, Journal of Network and Computer Applications 108 (2018) 87–111.
[16] Y.T. Jia, Q.Q. Zhang, W.N. Zhang, X.B. Wang, CommunityGAN: community detection with generative adversarial nets, in: Proceedings of the 28th International Conference on World Wide Web (WWW), ACM, 2019, pp. 784–794.
[17] D. Jin, J. He, B.F. Chai, D.X. He, Semi-supervised community detection on attributed networks using non-negative matrix tri-factorization with node popularity, Frontiers of Computer Science 15 (2021) 154324.
[18] D. Jin, Z.Y. Liu, W.H. Li, W.X. Zhang, Graph convolutional networks meet markov random fields: semi-supervised community detection in attribute networks, in: in: Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI), 2019, pp. 152–159.
[19] D. Jin, Z.Z. Yu, P.F. Jiao, S.R. Pan, D.X. He, J. Wu, P. Yu, W.X. Zhang, A survey of community detection approaches: from statistical modeling to deep learning, IEEE Transactions on Knowledge and Data Engineering (Early Access) (2021), https://doi.org/10.1109/TKDE.2021.3104155.
[20] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: Proceedings of the 5th International Conference on Learning Representations (ICLR), 2017, pp. 1–14.

[21] T.N. Kipf, M. Welling, Variational graph auto-encoders, in: Proceedings of the 30th Conference in Neural Information Processing Systems (NeurIPS), 2016, pp. 1–3.
[22] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Physical Review E 78 (2) (2008) 046110.
[23] X. Li, Y. Wu, M. Ester, B. Kao, X. Wang, Y.D. Zheng, Semi-supervised clustering in attributed heterogeneous information networks, in: Proceedings of the 26th International Conference on World Wide Web (WWW), ACM, 2017, pp. 1621–1629.
[24] X. Liu, W.J. Wang, D.X. He, P.F. Jiao, D. Jin, C.V. Cannistraci, Semi-supervised community detection based on non-negative matrix factorization with node popularity, Information Sciences 381 (2017) 304–321.
[25] M.E.J. Newman, Modularity and community structure in networks, in: Proceedings of the national academy of sciences 3 (23) (2006) 8577–8582.
[26] T. Pourhabibi, K. Ong, B.H. Kam, Y.L. Boo, Fraud detection: a systematic literature review of graph-based anomaly detection approaches, Decision Support Systems 133 (2020) 113303.
[27] H.H. Qiao, Z.H. Deng, H.J. Li, J. Hu, Q. Song, L. Gao, Research on historical phase division of terrorism: an analysis method by time series complex network, Neurocomputing 420 (2021) 246–265.
[28] Y.Y. Ruan, D. Fuhry, S. Parthasarathy, Efficient community detection in large networks using content and links, in: Proceedings of the 22nd International Conference on World Wide Web (WWW), 2013, pp. 1089–1098.
[29] L. Ruiz, F. Gama, A. Ribeiro, Gated graph recurrent neural networks, IEEE Transactions on Signal Processing 68 (2020) 6303–6318.
[30] O. Shchur, S. Günnemann, Overlapping community detection with graph neural networks, in: Proceedings of the 1st International Workshop on Deep Learning for Graphs (DLG), ACM, 2019, pp. 1–7.
[31] H. Shi, H.Z. Fan, J.T. Kwok, Effective decoding in graph auto-encoder using triadic closure, in: Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI), 2020, pp. 906–913.
[32] X. Su, S. Xue, F.Z. Liu, J. Wu, J. Yang, C. Zhou, W.B. Hu, C. Paris, S. Nepal, D. Ji, Q.Z. Sheng, P.S. Yu, A comprehensive survey on community detection with deep learning, IEEE Transactions on Neural Networks and Learning Systems (Early Access) (2022), https://doi.org/10.1109/TNNLS.2021.3137396.
[33] X.Y. Teng, J. Liu, M.M. Li, Overlapping community detection in directed and undirected attributed networks using a multiobjective evolutionary algorithm, IEEE Transactions on Cybernetics 51 (1) (2021) 138–150.
[34] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, in: Proceedings of the 6th International Conference on Learning Representations (ICLR), 2018, pp. 1–12.
[35] X. Wang, D. Jin, X.C. Cao, L. Yang, W.X. Zhang, Semantic community identification in large attribute networks, in: Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI), 2016, pp. 265–271.
[36] C. Wang, S.R. Pan, R.Q. Hu, G.D. Long, J. Jiang, C.Q. Zhang, Attributed graph clustering: a deep attentional embedding approach, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), 2019, pp. 3670–3676.
[37] M.J. Wang, L.F. Yu, D. Zheng, et al. Deep graph library: towards efficient and scalable deep learning on graphs, (2019) arXiv:1909.01315.
[38] W.J. Wang, X. Liu, P.F. Jiao, X. Chen, D. Jin, A unified weakly supervised framework for community detection and semantic matching, in: Proceedings of the 22nd Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), Springer, 2018, pp. 218–230.
[39] Z.H. Wu, S.R. Pan, F.W. Chen, G.D. Long, C.Q. Zhang, P.S. Yu, A comprehensive survey on graph neural networks, IEEE Transactions on Neural Networks and Learning Systems 32 (1) (2021) 4–24.
[40] J.R. Xie, S. Kelley, B.K. Szymanski, Overlapping community detection in networks: the state-of-the-art and comparative study, ACM Computing Surveys 45 (4) (2013) 1–35.
[41] Z.Q. Xu, Y.P Ke, Y. Wang, H. Cheng, J. Cheng, GBAGC: a general bayesian framework for attributed graph clustering, ACM Transactions on Knowledge Discovery from Data 9 (2014) 1-43.
[42] K.Y.L. Xu, W.H. Hu, J. Leskovec, S. Jegelka, How powerful are graph neural networks? in, in: Proceedings of the 7th International Conference on Learning Representations (ICLR), 2019, pp. 1–17.
[43] L. Yang, X.C. Cao, D. Jin, X. Wang, D. Meng, A unified semi-supervised community detection framework using latent space graph regularization, IEEE Transactions on Cybernetics 45 (11) (2015) 2585–2598.
[44] X.H. Yang, W.F. Liu, W. Liu, D.C. Tao, A survey on canonical correlation analysis, IEEE Transactions on Knowledge and Data Engineering 33 (6) (2021) 2349–2368.
[45] J. Yang, J. Leskovec, Overlapping community detection at scale: a nonnegative matrix factorization approach, in: Proceedings of the 6th ACM International Conference on Web Search and Data Mining (WSDM), ACM, 2013, pp. 587–596.
[46] L. Yao, C.S. Mao, Y. Luo, Graph convolutional networks for text classification, in: Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI), 2019, pp. 7370–7377.
[47] Y. Ye, S.H. Ji, Sparse graph attention networks, IEEE Transactions on Knowledge and Data Engineering (Early Access) (2021), https://doi.org/10.1109/TKDE.2021.3072345.
[48] Z.W. Zhang, P. Cui, W.W. Zhu, Deep learning on graphs: a survey, IEEE Transactions on Knowledge and Data Engineering 34 (1) (2022) 249–270.
[49] J. Zhou, G.Q. Cui, S.D. Hu, Z.Y. Zhang, C. Yang, Z.Y. Liu, L.F. Wang, C.C. Li, M.S. Sun, Graph neural networks: a review of methods and applications, AI Open 1 (2020) 57–81.