Neural Computing and Applications (2023) 35:24493–24511
https://doi.org/10.1007/s00521-023-09052-4

ORIGINAL ARTICLE

DAC-HPP: deep attributed clustering with high-order proximity preserve

Kamal Berahmand[1] · Yuefeng Li[1] · Yue Xu[1]

Received: 20 September 2022 / Accepted: 11 September 2023 / Published online: 3 October 2023

Abstract
Attributed graph clustering, the task of grouping nodes into communities using both graph structure and node attributes, is a fundamental problem in graph analysis. Recent approaches have utilized deep learning for node embedding followed by conventional clustering methods. However, these methods often suffer from the limitations of relying on the original network structure, which may be inadequate for clustering due to sparsity and noise, and using separate approaches that yield suboptimal embeddings for clustering. To address these limitations, we propose a novel method called Deep Attributed Clustering with High-order Proximity Preserve (DAC-HPP) for attributed graph clustering. DAC-HPP leverages an end-to-end deep clustering framework that integrates high-order proximities and fosters structural cohesiveness and attribute homogeneity. We introduce a modified Random Walk with Restart that captures k-order structural and attribute information, enabling the modelling of interactions between network structure and high-order proximities. A consensus matrix representation is constructed by combining diverse proximity measures, and a deep joint clustering approach is employed to leverage the complementary strengths of embedding and clustering. In summary, DAC-HPP offers a unique solution for attributed graph clustering by incorporating high-order proximities and employing an end-to-end deep clustering framework. Extensive experiments demonstrate its effectiveness, showcasing its superiority over existing methods. Evaluation on synthetic and real networks demonstrates that DAC-HPP outperforms seven state-of-the-art approaches, confirming its potential for advancing attributed graph clustering research.

Keywords Attributed networks · Attributed graph clustering · Deep clustering · Random walk with restart

1 Introduction

Recently, complex networks have been a prevalent tool to model many real-world entities and their relationships, such as online social networks, collaboration networks, citation networks, protein-protein interaction networks, and so on [1]. One fundamental component of these real networks is their underlying communities. Generally, a community is usually regarded as a group of nodes that are closely connected internally, whereas the external links between different communities are sparse. A network may contain several communities. The task of community detection [2] aims to find all these communities, which is also called graph clustering [3].

Traditional community detection algorithms rely solely on network topology to find communities. However, as the scale of networks grows, data sparsity difficulties in the network architecture may arise. It is insufficient to rely just on network topological structure to locate communities [4]. Consequently, most researchers exploit the attributed properties of nodes to compensate for structural sparsity in the real-world social network. In recent years, there has been an explosion of algorithms for attributed graph clustering that combine network topology and attribute features [5].

✉ Kamal Berahmand
kamal.berahmand@hdr.qut.edu.au

Yuefeng Li
y2.li@qut.edu.au

Yue Xu
yue.xu@qut.edu.au

[1] School of Computer Science, Faculty of Science, Queensland University of Technology (QUT), Organization, Brisbane, Australia

Attributed graph clustering is a new area that has lately received a lot of attention [6, 7]. The goal of attribute-based graph clustering is to put graph nodes into separate groups based on their attributes. Attributed graph clustering methods are divided into three methods traditional clustering, non-negative matrix factorization (NMF), and deep clustering. In traditional clustering, structure and attributes are combined as a pre-processing step, and after basic community detection methods, they are used to identify clusters [8]. Another method is the non-negative matrix factorization (NMF) method. NMF is a matrix method that consists of approximating a non-negative matrix of high rank by a product of non-negative matrices of lower rank, using the Frobenius norm as the approximation error [9]. NMF has been widely used in attributed graph clustering [10]. In this method, the adjacency matrix and the attribute matrix, which represent the node structure and node information of the network, are combined to form a hybrid matrix of structure and attributes that, after NMF obtain communities. Used to refer to methods. Traditional methods and NMF cannot be used for large data sets due to their high complexity. In contrast to these methods, deep clustering methods have been proposed which, using graph embedding procedures, represent each node in the network as a low-dimensional continuous vector such that main network information is efficiently encoded [11].

Deep learning advancements have recently ushered in a paradigm shift in artificial intelligence and machine learning, with astounding success on a wide range of tasks, including clustering. As a result, deep clustering has received a lot of attention [12, 13]. Deep clustering employs the deep learning method, followed by a nonlinear dimensional reduction to obtain high-discriminative features and the adoption of the node clustering algorithm for this new feature space. Deep clustering is divided into two categories based on whether these two steps are performed separately or simultaneously with a cost function: the two-phase and one-phase methods [14].

The two-phase group acquires communities in two stages. The embedding vector of each node is first learned using a network embedding technique. Second, the embedding vectors are grouped together using a traditional clustering technique like K-means or spectral clustering. This two-staged method is perfect for combining the benefits of network embedding with traditional clustering methods. The disadvantage is that the node clustering task is not helpful for the graph embedding learning and may not be the best match for the learned embedding for the future node clustering task. The one-phase method integrates new representation vectors and community detection into a single stage. Each node's new representation and community membership vectors are optimized and output simultaneously. The one-phase deep clustering has the potential to produce better results than two separate steps: During optimization, improved representation layer is learned to improve the cluster layer; the cluster layer, in turn, provides information to enhance the embedding [15]. However, most existing joint algorithms fail to integrate the global information of topologies and attributes into low-dimensional vector spaces. In complex attributed networks, inaccurate community detection may result from not taking the above factors into consideration.

In general, the deep-attributed graph clustering problem has three major obstacles.

- Most of them use two separate methods to do node representation learning and clustering. As a result, the networks' performance is limited since node clustering cannot be trained entirely.
- The majority of them don't provide the best representations of nodes because they don't reconstruct both node attributes and graph structure simultaneously. Most of them don't give the best representations of nodes because they don't at the same time.
- Prior research on attribute similarity focused mostly on the microscopic structure of networks and ignored global trends. So, the learned representations can't be used well for tasks like attributed graph clustering.

This paper presents an improved input of deep clustering for community detection, which overcomes the drawbacks of existing deep clustering-based community detection methods in attributed networks. We merge structural and attribute node information in a global, simultaneous, and integrated manner. We utilize a personalized random walk with a restart, which generates the matrix in some steps by integrating structural information and node attribute. The final matrix is created by combining these matrices. Then using deep join clustering to find the clusters. The authors evaluated the effectiveness of the proposed model using a variety of synthetic and real-world attributed networks; the experimental results show that our model performs better than traditional methods.

Our main contributions are summarized as follows:

- We propose a novel end-to-end approach called Deep Attributed Clustering with High-order Proximity Preserve (DAC-HPP) for attributed networks. This method jointly optimizes embedding learning and node clustering within a unified framework, resulting in mutual benefits for both components.
- To integrate the structural and attribute information of the network globally, the DAC-HPP algorithm constructs a consensus matrix by leveraging a random walk with restart method, facilitating the fusion of structure and attribute information.

- In order to capture long-distance similarities and enhance the topological structure, we introduce the concept of high-order attribute proximity.
- Through extensive testing on various synthetic and real-world attributed networks, our proposed approach demonstrates significant performance improvements compared to seven state-of-the-art methods tailored for attributed networks.

The paper is organized as follows. Section 2 provides an overview of traditional attributed graph clustering and deep attributed clustering. Section 3 covers the preliminaries of our work, including key concepts such as Random Walk with Restart, High-Order Structure Proximity, deep autoencoder, consensus matrix, and the proposed algorithm DAC-HPP. In Sect. 4, we present extensive experimental evaluations of our method on synthetic and real-world networks, demonstrating its effectiveness compared to existing techniques. Section 5 is the Discussion section, where we analyze and interpret the experimental results, discussing the implications, strengths, and weaknesses of our approach. Finally, in Sect. 6, we conclude the paper by summarizing the key contributions of our research.

2 Related work

Attributed graph clustering represents one of the most fundamental and actively researched problems in machine learning. Throughout the years, numerous approaches have been proposed to tackle the challenge of clustering attributed graphs, resulting in a wide range of methodologies and techniques [5, 16, 17]. These approaches can be categorized into three main groups: traditional clustering methods, nonnegative matrix factorization techniques, and deep clustering-based approaches.

2.1 Traditional method

In traditional methods, the initial step is to develop a way to fuse structural and attribute data, after which a community detection method is applied. Weigh-based methods are well-known instances of these categories, in which topological and attributed-based similarities are combined and assigned to the edges of the original graph, transforming the graph into a weighted graph. The modified graph can then be used in conjunction with any traditional community detection method [18, 19]. This study [20] assumes that in a network, there may be certain nodes with similar attributes, and as a result, they tend to belong to the same community. Despite this, they are not connected by a link due to several circumstances, such as sparsity. As a result, they proposed augmenting the original network by adding a k-Nearest Neighbour (kNN) graph containing node attributes. The community structure of the improved network is then extracted using a clustering method, including K-means. Li et al. [21] have suggested an overlapping technique that uses a combination of local and global information about nodes in the network structure. An enhanced version of PageRank is used to provide global information. To find overlapping communities, the structural information and node attributes are combined into forms of node convergence degrees.

Proposed weight modification approach (PWMA) and proposed linear combination approach (PLCA) approaches have been proposed by Alinejad et al. [22]. The original attributed network is fed into PWMA, which transforms it into a non-attributed network. In this phase, similarity measures (Jaccard, cosine, and angular) would be used; in this study, they are Jaccard, cosine, and angular. The second phase is to detect communities in the constructed secondary network using mixed-integer linear programming (MILP) method. PWMA does not add or remove edges from the original network, only the weights change. The idea behind PLCA [23] is that by utilizing a linear combination of network topology and attributes, nodes with a similarity higher than the desired threshold but that are structurally divided can be held close together. The transformation phase to the secondary network is also included in PLCA, but its edge set may differ from that of the source network and edges may be added.

2.2 Nonnegative matrix factorization

Nonnegative matrix factorization methods: NMF is a low-rank matrix factorization model that has been around for a long time [24]. Given a nonnegative matrix $X = [X_1, X_2, .., X_n]$, which is formed from a collection of n m-dimension data vectors and a desired reduced dimension d (such that $d \leq min(m, n)$), The goal of NMF is to discover two nonnegative matrices $W = [W_{ip}]^{m\times d} \in \mathbb{R}_+^{m\times d}$ and $H = [H_{ip}]^{n\times d} \in \mathbb{R}_+^{n\times d}$, which can well approximate the original matrix X in the form of their product:

$$X = WH^T \tag{1}$$

where W and H are the basis matrix and coefficient matrix, respectively. In order to estimate the factorization matrices, it gets the following objective function:

$$\min_{W,H} \ \ell(W, H) = \|X - WH^T\|_F^2 \quad \text{s.t.} \quad W \geq 0, H \geq 0. \tag{2}$$

The non-negativity constraints on both factor matrices result in parts-based representation, which is compatible with the notion of learning the parts in order to form the whole. The conventional NMF model: $X{=}WH^T$ can be

directly used to detect communities by substituting X with A. Each column of the matrix W corresponds to a community representation. Furthermore, each column of matrix H represents the membership between overall pairs of communities and nodes. As a result, it can assign a community label to the node by determining the index of the biggest element in the coefficient matrix, and the NMF detection rule is:

$$\text{community}(v_j) = \arg \max_{i=1,2,\ldots,V_{j1}} \tag{3}$$

Different versions of NMF have been introduced to attributed graph clustering in recent years. Ye Li proposed [25] a pioneering approach for community detection that effectively incorporates both community structures and node attributes. The method introduces a community structure embedding technique, which encodes inherent community structures based on underlying community memberships. By leveraging the information from both node attributes and the community structure embedding, the method formulates attributed community detection as an optimization problem using nonnegative matrix factorization. Xiao Wang [26] introduced the Semantic Community Identification (SCI) method, a novel approach for attributed graph clustering that combines network topological and node semantic information. SCI utilizes nonnegative matrix factorization (NMF) to integrate topology-based community memberships and node attribute-based community attributes. The fundamental idea behind SCI is that nodes with similar community memberships are more likely to be connected, while nodes with attributes consistent with the underlying community attributes are more likely to belong to the same community. Li et al. [27] proposed an embedding-based method using NMF in which the node attributes and community structure embedding were used to detect communities of the network. The proposed study gives two joint nonnegative matrix factorization algorithms that don't need any parameters. These algorithms can find community structures in attribute networks.

Jianyong [28], a new graph neural network encoding method is suggested for attribute graph clustering. Each edge in an attribute network is connected to a continuous variable based on the encoder. A continuous-valued vector is transformed by nonlinear transformation into a discrete-valued community grouping solution. To assess the attribute homogeneity of nodes in communities, two objective functions for single-attribute and multi-attribute networks are proposed. The methods integrate topology and attribute data to produce more sensible and understandable results. The study's authors [29] propose a nonlinear NMF-based model called NMFGAAE, which aims to improve performance by combining NMF community detection with graph attention autoencoder models.

2.3 Deep clustering-based

In graphs, deep clustering algorithms can capture complex and nonlinear node properties [30]. For this purpose, several recent papers have introduced deep clustering methods into graph-attributed clustering tasks with the goal of learning useful node embeddings for clustering. These studies use both two-phase and one-phase modelling methods [31, 32]. Feature learning and clustering are regarded as upstream and downstream tasks in the two-phase technique. One-phase methods use standard clustering algorithms like k-means and spectral clustering along with the node embedding that was learned by the deep learning method to make clusters.

In this study [33], the representation mechanism is based on autoencoders that transform attributes into node embedding vectors based on network topology using an adversarial learning algorithm. ANRL [34] presents a neighbour improvement autoencoder that reconstructs its target neighbours rather than itself in order to represent node attribute data. The attribute encoder is used to create an attribute-aware Skip-gram model that describes the relationships between each node and its direct or indirect neighbours. In order to capture high nonlinearity while preserving the topological structure and node attribute proximities, DANE [35] uses a deep attributed network modelling. Both topological structure and node attributes are encoded into low-dimensional vectors using two autoencoders.

The one-phase methods create an end-to-end framework that may directly get cluster results by integrating feature learning with clustering. Using these methods, the discrete clustering issue is often converted into a differentiable optimisation problem and integrated into a single framework [36]. DNENC (Deep Neighbour-aware Embedded Node) [37] performs end-to-end graph embedding and node clustering using an unsupervized deep neighbour-aware embedding algorithm. The topological structure and node content of a graph are encoded into a compact representation using a neighbour-aware graph autoencoder that gradually integrates information from neighbours using an attentional or convolutional encoder. In this study, a SENet (spectral embedding network) [38] for attributed graph clustering is proposed. By including the knowledge of common neighbours, the noisy and sparse graph structure may be significantly enhanced. By learning node embeddings in response to a spectral clustering loss, information about the global cluster structure can be explicitly or implicitly added to the node embeddings of different layers.

3 Proposed method

In this section, we elaborate on the DAC-HPP (Deep Attributed Clustering with High-order Proximity Preserve) method, specifically designed for deep attributed graph clustering with a focus on higher similarity. Firstly, we provide an overview of the network formulation and key definitions. Subsequently, we explain the DAC-HPP method, which consists of two essential steps. The first step involves constructing a consensus matrix that combines both structural and attribute information to create proximity matrices, with a particular emphasis on high-order similarity. We then integrate these proximity matrices into the input for the end-to-end clustering framework. In the second step, we employ an end-to-end framework to discover clusters. This framework includes a layer representation comprising a deep autoencoder, along with a clustering layer incorporating a self-supervisor. Detailed explanations of our proposed model will be presented in subsequent sections.

3.1 Problem formulation

We define an attributed network as $G = (V, E, T)$ where $V = \{V_1, V_2, \ldots, V_n\}$ and $E = \{E_1, E_2, \ldots, E_m\}$ and $T = \{t_1, t_2, \ldots, t_t\}$ represent nodes, edges, and node attributes, respectively. The adjacency matrix A for G materializes the edges between nodes of the graph in the form of the matrix such that the A_{ij} contains the value of 1 if v_i and v_j have a common edge and 0 otherwise. Our objective is to partition the graph's nodes G into k clusters $C = \{C_1, C_2, \ldots, C_k\}$. Additionally, attributed graphs are considered of as undirected, connected, and simple graphs in which each node attribute fits a unique multi-dimensional schema. The various notations used in this paper are listed in Table 1.

Table 1 Notations and explanation

The given complex network	G
Nodes set	V
Edges set	E
Attribute set	T
The number of nodes	n
The number of communities	k
The embedded features vector	Z
The adjacency matrix	$A \in \mathbb{R}^{n \times n}$
The attributed matrix	$T \in \mathbb{R}_{+}^{n \times t}$
The proximity matrix	$P \in \mathbb{R}_{+}^{n \times t}$
The consensus matrix	$C \in \mathbb{R}^{n \times n}$
The representation loss	L_{rep}
The clustering loss	L_{clu}

Definition 1. Random walk with restart (RWR): Many graph mining applications like link prediction, recommendation, anomaly detection, semi-supervized learning, and network embedding depend on the ability to calculate effective node-to-node similarity [39]. RWR [40] is a useful node-to-node relevance score that takes into account global network topology [41] and detailed edge relationships [42]. RWR calculates the proximity between two nodes by exploring the global network topology many times. Starting at a seed node, the random walker distributes its resources by (1) moving to one of its surrounding nodes with probability $1 - c$ and (2) restarting with probability c from the seed node. This method is continued until all nodes have been visited. The probability vector created at this point comprises the proximity scores of all nodes as well as the seed node. The equation (4) represents the proximity scores of all nodes throughout each step.

$$p = (1 - c)Wp + cq \tag{4}$$

where q is a vector with a seed node is set to 1 while the remaining elements are set to 0, W is network adjacency matrix, which represents the matrix of node transitions, and $W_{i,j}$ reflects the probability of transitioning from node j to node i. c is set to 0.8 according to Woojeong et al. [41] work. After several repetitions, p would attain a stable state. As a result, it diffuses its resources throughout the network by multiplying the adjacency matrix. The proximity score matrix P can be made by iterating the p_i value, which is the result of RWR for the seed node i. Each entry in the matrix shows how close node j is to the seed node i.

Definition 2. High-order structure proximity: In terms of structure proximity, two nodes are the same if they are connected and have the same k-order neighbours. The adjacency matrix represents the first-order proximity, which describes the local pairwise similarity between vertices. In most cases, the first-order proximity matrix is insufficient to adequately capture the pairwise proximity between vertices [43, 44]. As a result, higher-order proximity is used to describe vertices' strength. The number of shared neighbours between vertices, for example, can be used to define second-order proximity. The probability that a 2-step random walk from v_i reaches v_j may also be used to represent the second-order proximity between v_i and v_j. If v_i and v_j have a lot of shared neighbours, the probability will be high. We can easily generalize the probabilistic setup based on a random walk to k-order proximity: the probability that a random walk starts at v_i and goes

approximately k steps to v_j. The transition probability matrix of a single-step random walk is the normalized adjacency matrix A. The k-step transition probability matrix may thus be computed as the k-order proximity matrix

$$A^k = \underbrace{A \cdot A \cdot A \cdots A}_{} \tag{5}$$

where A_{ij}^k represents the k-order proximity between vertex v_i and v_j.

Definition 3. High-order attribute proximity: Smoothing a data set is an approximation function in image processing that aims to capture essential patterns in the data while ignoring noise or other fine-scale structure phenomena. As a result, we developed the attribute smoothing filter with topological high-order neighbours, which preserves the intrinsic properties of the attribute while reducing noise. The mean filter is a suitable choice for our deep clustering tasks. The average value of neighbour nodes is calculated using the mean filter. An averaged smoothness attribute value can well represent the pairwise similarity nodes in a k-order neighbour, which is precisely what we want the attribute homogeneity in clustering to achieve [45]. The size of the filter window is something to consider. The value of neighbour nodes is calculated via smoothing filtering, and the number of neighbour nodes is proportional to the smoothing effect. The number of high-order neighbours was used to determine the window size.

We can increase the smoothing effect by expanding the range of neighbours in this way. More information from neighbours can be introduced through a bigger window. We may also use high-order proximity to interpolate global topological relations while conserving local structure information. High-order proximity can direct the filtering process when using high-order neighbours as the filter window. As a result, we define the attribute smoothing filter for node i with high-order topological neighbours as the following:

$$smooth_{N_{(h)},i} = \frac{\sum_1^{|N_h(i)|} T_j}{|N_h(i)|} \tag{6}$$

in which $|N_h(i)|$ is the high-order neighbours of node i.

After computing the attribute smoothing filter using the high-order neighbour, we'll use cosine similarity to determine how similar each node is to its neighbours. Therefore, we used the influence of the high-order neighbour to consider the similarity of each node to its neighbour node. The similarity pair nodes v_i and v_j from the attribute perspective, taking into account the high-order neighbour, will be defined as follows:

$$S(i,j) = \frac{\mathrm{Sum}(T_i \cdot T_j)}{\sqrt{\mathrm{Sum}(T_i \cdot T_i)}\sqrt{\mathrm{Sum}(T_j \cdot T_j)}}, \tag{7}$$

where T_i and T_j are the one-hot attribute vector representation of node v_i and v_j, respectively.

Definition 4. Deep autoencoder: Deep learning is a subfield of machine learning based on multi-level data representation learning, in which low-level features are passed up through layers to higher-level features. These deep architectures have made significant progress in the field of deep clustering by automatically learning essential features from images, text, or graph data. In recent years, the autoencoder (AE) algorithm and its deep version (DAE), like classic dimensionality reduction methods, have had a lot of success [46]. A single autoencoder (AE) consists of a two-layer neural network consisting of an encoder and a decoder.

Let $a_i \in R^N$ be an adjacency vector of A that contains the ith node's local neighbourhood. The autoencoder architecture consists of a collection of nonlinear transformations on a_i split down into separate parts: encoder $g(a_i)$: $R^D \rightarrow R^N$ and decoder $f(g(a_i))$: $R^N \rightarrow R^D$ The encoder and decoder's hidden representations are calculated as follows:

$$\begin{aligned} &Encoder \quad z_i = g(a_i) = f(W_e x_i + b_1) \\ &Decoder \quad a_i = a_i = f(z_i) = f(W_d x_i + b_2) \\ &autoencoder \quad a_i = h(a_i) = f(g(a_i)) \end{aligned} \tag{8}$$

in which f is a nonlinear activation function like sigmoid or ReLU. W_e is encoder weights, W_d is decoder weights, b is the bias parameters.

In some datasets, the number of input features may surpass the capacity of a single autoencoder and have a complex relationship. Consequently, the use of a single autoencoder is inadequate. Therefore, a stack autocorrelator is used in these situations. As an input feature, it provides the $(k+1)th$ layer, which is the hidden layer of the kth AE. A crucial feature of stacked autoencoders is unsupervized pre-training, layer by layer as input is processed. The first layer can be utilized as an input for the following autoencoder once it has been pre-trained. Backpropagation can be used to fine-tune a neural network that has already been trained. The last layer may be used for traditional supervized classification.

3.2 Constructing consensus matrix

Creating an accurate input matrix is a critical challenge in deep clustering as it significantly impacts the output of the learning framework. Deep clustering aims to achieve more precise clusters by incorporating higher-order proximity information, as relying solely on first-order or second-order

proximity may not fully capture the underlying relationships between nodes. In traditional clustering methods, first-order proximity is often represented using pairwise similarity or distance measures, but this approach may struggle to capture the complex relationships present in real-world datasets. Second-order proximity, which considers pairwise relationships between neighbouring nodes, provides a more detailed representation but still has limitations in capturing the global structure of the data. On the other hand, higher-order proximity takes into account multi-step relationships and offers a more comprehensive view of the data structure. By integrating high-order structural and attribute information, we can better capture the intricate relationships and dependencies within the data, leading to improved clustering accuracy. Thus, the use of high-order structural and attribute information is crucial in overcoming these limitations and enhancing clustering performance.

In this section, we propose an approach that addresses these challenges by integrating high-order structural and attribute information into a deep clustering framework. Our method consists of two distinct components, each contributing to the integration process. The first component utilizes a personalized random walk with restart technique, incorporating variable step lengths, to generate a proximity matrix. This personalized random walk effectively captures the intricate relationships within the graph by considering both structural and attribute similarities. The second component involves combining the generated proximity matrix with another matrix to construct a consensus matrix. This consensus matrix aggregates information from multiple sources, yielding a unified representation that encompasses both structural and attribute characteristics. In the subsequent sections, we will delve into detailed explanations of each component, elucidating their respective roles in enhancing the integration of high-order structural and attribute information within our proposed deep clustering framework.

3.2.1 Step-based proximity matrix calculation

We construct a step-based proximity transition matrix, denoted as P^t, by performing a random walk on the network. This matrix captures the proximity between nodes based on the t-degree of proximity, taking into account both the structure matrix A and the attribute proximity matrix S at each step t. In terms of the structure random walk, if there is a link between a pair of nodes (v_i, v_j), it indicates their high similarity, implying that node v_i will likely transition to node v_j with a high probability. Similarly, in terms of attribute proximity, if a pair of nodes (v_i, v_j) share many common attributes, they are expected to have a closer relationship.

To capture the best similarity between node pairs (v_i, v_j) based on both structural and attribute information, we define a personalized random walk with restart. In this process, a seed node v_i transitions to the next node based on the structure matrix A with probability α, and hops to other nodes based on the attribute proximity matrix S with probability $1-\alpha$. The first-degree link between nodes v_i and v_j is represented as follows:

$$\mathbf{P}_{ij}^1 = \begin{cases} \alpha A(i,j) + (1-\alpha)S_{ij} & \text{if } e_u, e_v \in E \\ (1-\alpha)S_{ij} & \text{otherwise} \end{cases} \tag{9}$$

where A is the structure matrix containing the transition probabilities between nodes within one step, and S is the attribute proximity matrix containing the same information. Once this step is completed, we can use the t-th proximity matrix P_t to simulate the (t+1)-th step-based proximity for any $t \geq 1$ as follows:

$$P^{t+1} = \alpha P^t A + (1-\alpha)S^t \tag{10}$$

In the above equation, the proximity at the (t+1)-th step is obtained by multiplying the t-th step proximity matrix P_t with the graph structure matrix A, weighted by α. The closeness of node attributes at the (t+1)-th step is captured by S^t, as node attributes are dynamic and change with the network structure. Finally, the t-th structure proximity and attribute proximity are combined to form the final (t+1)-th step proximity. It is expected that higher-order proximity will be more effective than lower-order proximity in capturing the complex relationships within the network.

3.2.2 Integrating high-order structural and attribute information

In this section, we propose a method for generating the consensus matrix by integrating multiple matrices that contain global structural and attribute information at different steps of a random walk restart. This integration process results in a final consensus matrix that effectively combines high-order structural and attribute information. To begin, we integrate the t-degree proximity matrix sequences $P_1, P_2, ..., P_t$ into the previously calculated consensus matrix $C \subseteq R^{|n|\times|n|}$. This consensus matrix preserves both the local and global structural information of the graph, as well as the node attribute information. To account for the varying importance of nodes with different levels of proximity, we utilize a weight function that monotonically decreases as step t increases. By applying this weight function, the proximity consensus matrix C is defined as the combination of different k-step data,

allowing us to capture the comprehensive global structural information of the graph.

In our approach, we assume a fixed length of random walk steps (t) to be 5. Additionally, we set the parameter β, which represents the importance of each matrix in the combination process, to 0.9. These values are based on previous research studies that have investigated optimal parameters for the consensus matrix.

The integration of the proximity matrices is mathematically expressed by the following equation:

$$C = \sum_{m=1}^{M} w(t).P^t \tag{11}$$

where the weight function $w(t)$ is defined as $w(t) = \beta^t$, with $\beta \in (0, 1)$. This equation captures the combination of different k-step data, allowing us to effectively capture the global structural information related to the graph. The creation of the consensus matrix involves two main components: proximity matrix calculation and the integration of the proximity matrices, as illustrated in Fig. 1.

3.3 The end-to-end framework of DAC-HPP

Attributed graph clustering aims to identify community structures in attributed networks, which poses challenges due to the requirement for an effective representation of both the network's topology structure and attributes in clustering analysis. In recent years, attributed graph clustering has evolved from traditional shallow methods to deep learning approaches, harnessing unique feature representations and the potential of deep learning. Deep clustering utilizes deep learning methods, followed by nonlinear dimensional reduction, to extract highly discriminative features and applies node clustering algorithms in this novel representation space.

To address these limitations and challenges, we propose DAC-HPP, a novel end-to-end deep attributed graph clustering method. DAC-HPP utilizes personalized random walk with restart to capture high-order similarities of structural and attribute information, enabling it to concurrently extract latent embeddings and predict clustering assignments. The method consists of two modules: representation learning and clustering, similar to existing deep clustering methods.

The representation learning module employs Graph Neural Networks (GNNs) to learn cluster-friendly node embeddings, while the clustering module estimates the number of clusters. As effective representations contribute to strong clustering and high-quality clustering provides supervisory signals for representation learning, these modules mutually support each other. In a unified framework, we jointly optimize the parameters of these components using the following equation:

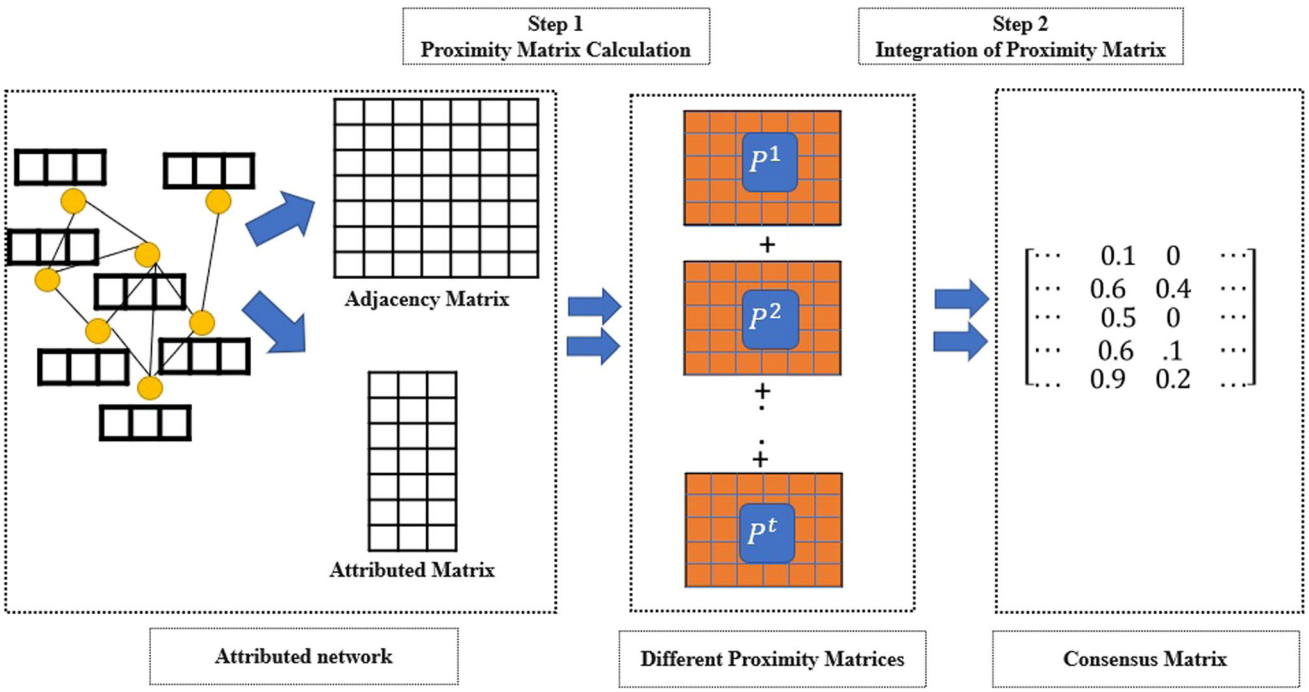

Fig. 1 The process of creating the consensus matrix, including the generation of random walks of different lengths and their combination

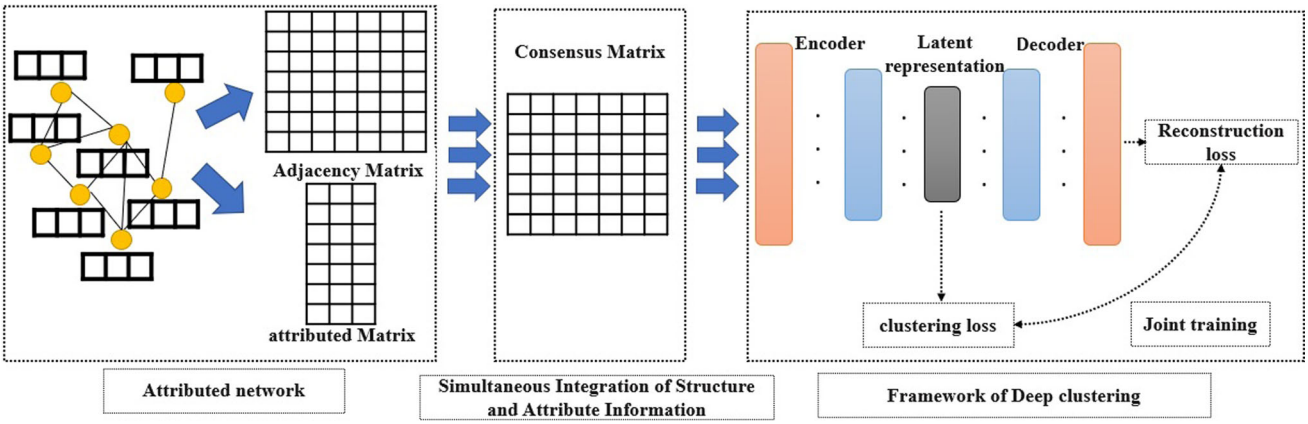

Fig. 2 The basic architecture of the model we propose the input adjacent matrix (A) and the attribute matrix (X) of an attributed graph are, respectively. Hidden layer outputs h is integrated into node embeddings Z via the encoder model. After that, the structural and content decoders reconstruct A and X, respectively. We use "A" and "X" to represent the decoder's output. To achieve self-separation regularization, the KL-divergence between node similarity distributions P and Q is minimized. P and Q are pairwise node embedding similarities represented by the student's t-distribution

$$L_{final} = L_{rep} + \alpha L_{clu} \tag{12}$$

where $\alpha \geq 0$ acts as a control coefficient to balance the representation module and clustering module, and L_{rep} and L_{clu} represent the objective functions of the representation learning phase and cluster estimation phase, respectively.

During the representation learning phase, the decoder reconstructs the attributes, denoted as $\widehat{X} = \widehat{Z}_a^l$. Consequently, the autoencoder's reconstruction loss is defined as:

$$L_{rep} = \frac{1}{2N} \|X - \widehat{X}\|_2^2 \tag{13}$$

After obtaining the latent representation $\widehat{Z}_a^l \in \mathbb{R}^{n \times d}$ from the node attributes, we use it to construct the soft cluster assignment $\mathbf{Q}_a \in \mathbb{R}^{n \times k}$, where q_a denotes the probability that node v_i is assigned to cluster k and is calculated using Eq. 14:

$$q_{ik}^a = \frac{(1 + (\|q_i^a - \mu_a^k\|^2)}{\sum_j (1 + (\|q_i^a - \mu_a^k\|^2)} \tag{14}$$

In Eq. (14), the Student's t-distribution is used as a kernel to measure the similarity between the node representation z_i and the clustering centre μ_k^a. The higher the probability, the closer the nodes are to the cluster centre, resulting in more accurate soft assignments. The target distribution P_a is then built using Eq. (15) to emphasize data points assigned with high certainty and improve cluster purity. It is important to note that the initial centres μ_k^a $(k = 1; 2; ...; k)$ are obtained by pre-training the autoencoder solely on reducing the reconstruction loss. After that, we initialize the learned representations using K-means and do not employ K-means clustering during the iterative training.

$$p_{ij} = \frac{q_{ij}^2 / \sum_i q_{ij}}{\sum_j (q_{ij}^2 / \sum_i q_{ij})} \tag{15}$$

In accordance with the other research [47], we aim to force the current assignment Q_a to converge to the goal distribution P_a, enabling for the joint training of representation and cluster assignments to produce the final clusters. The KL divergence between two distributions is used to define the clustering loss as shown below.

$$L_{clu} = Kl(Q\|P) = \sum_i \sum_j log\ t_{ij} \frac{q_{ij}}{p_{ij}} \tag{16}$$

We minimize Eq. (16) to get reliable clusters, and as Q_a reaches idempotence, it approaches 0. At this point, the cluster structure is obvious and the cluster distribution's entropy is low. The core idea of deeply embedded clustering, commonly known as "self-supervized," is presented in this approach [48]. The proposed algorithm comprises two primary components. The first component involves generating an input that incorporates global information regarding the structure and attributes. On the other hand, the second component is an end-to-end clustering framework, as illustrated in Fig. 2.

Algorithm 1 The proposed DAC-HPP deep clustering algorithm

Input: network G= (V, E, X), consensus matrix, number of clusters K, pre-training weights
Output: Attributed Graph Clustering $C = \{C_1, C_2, \ldots, C_k\}$
Initialization:
1: Assign a unique label to each node in the network
2: The set of cluster centroid μ
3: Import the pre-trained encoder weights W_e and decoder weights W_d to the network
Repeat:
4: Extract the embedded feature Z from the deep autoencoder pre-presentation learning module.
5: Update target distribution P in the clustering module through the formulate 15.
6: Save the current label assignment C as C_{old} , and compute the new label assignment C using the formula 16.
7: Select n samples from the input $G = (V, E, X)$ to form a batch B.
8: Calculate the loss of representation learning L_{rep} and clustering L_{clu}.
9: Update the cluster centroid μ, encoder weight W_e, and decoder weight W_d.
end while
Return Attributed Graph Clustering.

4 Experimental results

In this section, we conduct extensive experiments on synthetic networks and real-world networks to evaluate the effectiveness of DAC-HPP. All the experiments are performed on a processor: Core i7 @ 4.2 GHz, RAM: 16 GB, and a 64-bit Windows operating system. Additionally, we use the Python programming language to implement DAC-HPP.

The organization's structure is shown below Sect. 4.1 summarizes the datasets used in the following studies. In sect. 4.2, it is explained how many nodes should be in each layer of the autoencoder. SCNCD [49], SA-cluster [50], CDE [25], SCI [26], TASNMF [27], CE-MOEA [19], and SENet [38] are seven well-known and state-of-the-art comparison methods discussed in Sect. 4.3. The evaluation metrics are discussed in Sect. 4.4. The method's efficiency is examined on several types of synthetic datasets in Sect. 4.5. The detailed findings of the synthetic dataset were compared to five state-of-the-art methodologies based on three commonly used evaluation measures. The results of the performance evaluation on real-world datasets are presented in Sect. 4.6.

4.1 Datasets description

The detailed information on synthetic networks and real-world networks is as follows.

4.1.1 Synthetic dataset

LFR-EA is a synthetic network generated using the benchmark proposed by Elhadi and Agam [51], which is an extension of the LFR benchmark by Lancichinetti et al. [52]. This network generator utilizes two parameters, μ and v, both ranging from [0.1, 0.8], to control the structure and attribute values. The mixing parameter μ determines the ratio of intra- and inter-community connections. Lower values of μ result in a clearer community structure, with more intra-cluster links compared to inter-cluster links. On the other hand, v represents the noise attribute parameter, where lower values lead to similar features among nodes belonging to the same community. By combining different values of μ and v, graphs with varying levels of structural and attribute ambiguity can be generated.

A benchmark network consisting of 1000 nodes is generated, named LFR-EA-1000, to evaluate all aspects of

Table 2 LFR-EA-1000 parameters setting

Parameter	Values
Number of vertices (N)	1000
Average degree (k)	25
Maximum degree ($maxk$)	40
Mixing parameter (μ)	[0.1:0.8]
Exponent for the community size distribution ($\tau 1$)	1
Minimum for the community size ($minc$)	60
Maximum for the community size ($maxc$)	100
Number of attributes	10
Attribute's domain clusters assignment ($ainf$)	1
Attribute range (R)	15
Attribute noise ($ainf$)	[0.1:0.8]

DAC-HPP. A variety of instances of the parameter combination reported in Table 2 are generated. We take the average of the results from ten runs because making networks is a random process, and different runs may end up with different partitions.

4.1.2 Real-world dataset

In addition to synthetic networks, the proposed method is tested on six real-world attributed networks with known community structures.

The WEBKB dataset [53] consists of four webpage networks of students from four universities: Texas, Cornell, Washington, and Wisconsin, in which the students' websites correspond to nodes, and the links between them are considered edges. Also, the presence or absence of keywords in the webpages is content information of nodes, and there exist five communities of these webpages: course, faculty, student, project, and staff.

Cora [53] and PubMed [53] are citation networks in which vertices represent articles, and a link connects two articles provided that one of them cites the other. Keywords in articles play the role of nodal attributes. Table 3 summarizes the fundamental statistical properties of the used datasets.

Table 3 Dataset statistical properties

Datasets	|V |	|E|	NA	NC
Texas	187	328	1703	5
Cornell	195	304	1703	5
Washington	230	448	1703	5
Wisconsin	265	530	1703	5
Cora	2708	5429	1433	5
PubMed	19,717	44,338	500	3

Table 4 Neural network structures

Datasets	Layer configuration
Texas	N-128-64
Cornell	N-128-64
Washington	N-128-64
Wisconsin	N-256-128
Cora	N-2048-1024-512
PubMed	N-16,384-8192-4096-2048
LFR-EA	N-512-256-128

4.2 Parameter settings

The middle-hidden layer of the deep autoencoder is mostly used for cluster analysis. The deep neural network analysis was used to demonstrate that the findings for all datasets were consistent. The network was configured as a full autoencoder, with the dimensions selecting different layer number depths. In general, the community detection performance of the deep autoencoder structure with 3 layers is superior to that of the deep autoencoder structure with 2 layers. For various synthetic and real-world networks, the deep autoencoder structure in this study employs a varying number of stacked autoencoders. The layer settings for each dataset on the stacked autoencoder are detailed in Table 4.

Additionally, we configured the deep autoencoders to activate each layer using the sigmoid function. To train the model, we gave the deep autoencoders 10 random initializations and used the low-dimensional representation of the hidden layer as the output.

4.3 Baselines

This paper compares DAC-HPP with seven state-of-the-art attributed graph clustering methods. All the baselines are performed with default parameters, and their introductions are summarized as follows.

- SCNCD [49] came up with an overlapping community detection algorithm based on a spectral clustering algorithm that uses the degree of node and structure convergence by combining the attributes of nodes and the network's structure.
- SA-cluster [50] uses random walks to obtain a unified distance measure between nodes in an augmented network. The augmented graph captures the structural

and attribute-based proximities, and therefore the obtained distance can represent both similarities. Then, these similarities between node pairs are used by a clustering algorithm, i.e., K-medoids, to find the partitions.

- CDE [25] proposed an NMF-based embedding method to learn the network's community layout using structural and node attributes.
- SCI [26] proposed an NMF-based method that combines the nodes' topological and attributed community memberships into a single community structure.
- TASNMF [27] proposes a parameter-free joint nonnegative matrix factorization model that can integrate topology and attribute data from complex networks.
- CE-MOEA [19] proposes a nonlinear NMF-based model, which aims to improve performance by combining NMF community detection with graph attention autoencoder models.
- SENet [38] is a spectral embedding network for attributed graph clustering that enhances graph structure by utilizing the knowledge of common neighbours and determining node embeddings using a spectral clustering loss.

4.4 Evaluation metrics

In this paper, two types of assessment metrics are used to compare the quality of communities generated by different methods: quality-based and information-recovery-based metrics. In the first type, the quality of identified communities is assessed using the basic definition of community. On the other hand, information recovery-based metrics are based on network partitions' ground truth information.

4.4.1 Information recovery-based metrics

Let X and Y represent two different sets of discovered communities and ground-truth communities, respectively. Then x_i and y_i represent the ith community of these sets. To compare the two sets, we can use the Normalized Mutual Information (NMI) and the Rand Index (RI).

- **NMI** [54]: NMI can be used in clustering to measure the similarity of two clustering results and is an important measure of community discovery, which can basically evaluate the accuracy of a community division in relation to a standard division in a more objective way. The value range of NMI is from 0 to 1, and a higher number means that the results are closer. The NMI is defined by

$$\mathrm{NMI}(A,B) = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} C_{ij} log\frac{nC_{ij}}{C_i C_j}}{\sum_{i=1}^{C_A} C_i \cdot log\frac{C_i}{n} + \sum_{j=1}^{C_B} C_j \cdot log\frac{C_j}{n}}, \tag{17}$$

where $C_A(C_B)$ is the number of clusters in the sets of $A(B)$, $C_i(C_j)$ is the sum of the elements of C in $rowi(columnj)$, and n is the number of vertices. If the set of predicted clusters is the same as existing clusters, i.e. $A = B$, then $NMI = 1$, and in case they are completely different, $NMI = 0$.

- **RI** [55]: The Rand Index computes the percentage of node pairs that are correctly clustered. Suppose we have access to the ground truth clusters $C = \{C_1, C_2, C_3, \ldots, C_k\}$ and the obtained clusters by the clustering method $C' = \{C'_1, C'_2, C'_3, \ldots, C'_k\}$. Then suppose that G_1 represents the pair of nodes that are both present in the same clusters of C and C', G_2 represents the pair of nodes that are present in the same clusters of C, but not present in the same clusters of C', G_3 represents the pair of nodes that are not present in the same clusters of C, but present in the same clusters of C', and finally G_4 represents the pair of nodes that neither are both present in the same clusters of C, nor C'. Then, the RI measure can be calculated using the following equation:

$$RI = \frac{G_1 + G_4}{G_1 + G_2 + G_3 + G_4} \tag{18}$$

4.4.2 Quality-based metrics

In many networks, the lack of ground-truth communities has made comparing community detection approaches difficult. As a result, quality-based metrics are supplied to assess the quality of network partitioning based on network definitions. In this paper, we use modularity, a quality-based criterion, to look at how well-discovered communities are doing.

- **Modularity** [56]: Newman first proposed the definition of modularity to measure the goodness of community discovery algorithms. One criterion for judging whether a community discovery result is reasonable is by determining whether the nodes are tightly connected within the community, while the connections between different communities are sparser, and communities that satisfy this condition will have a larger modularity. Nowadays, equation (19) is commonly used to calculate modularity.

$$Q(X) = \frac{1}{2\|E\|}\sum_{ij}\left(A_{ij} - \frac{K_i K_j}{2\|E\|}\right)\delta_{ij}, \tag{19}$$

where k_i denotes the node's degree i and δ_{ij} the Kronecker delta function, as follows:

$$\begin{cases} \delta_{ij} = 1, & v_i \text{ and } v_j \text{ are in the same community,} \\ 0, & \text{otherwise.} \end{cases}$$

4.5 Evaluation of synthetic networks

We first compare the performance of algorithms on LFR-EA networks by using NMI, RI, and modularity.

In this section, we evaluate the performance of the comparison methods on synthetic networks in terms of both information recovery and quality-based metrics during two experiments. Eight networks were created for this experiment with the parameters $N = 1000$, $k = 25$, $max - k = 40$, $min - c = 60$, $max - c = 100$, $nattr = 10$ (int $attr$ [1, 15]), $v = 0.5$, and the mixing parameters [0.1, 0.8] to examine how well various methods performed in networks with various resolutions of the community structure controlled by μ. It should be noted that the visibility of distinct community structures decreases as the value of the μ parameter increases.

The findings shown in Fig. 3 provide evidence to support this claim. This figure compares the NMI performance of DAC-HPP with that of other methods on these networks. According to the results in Fig. 3, the majority of the methods perform well for small values of the mixing parameter $(0.1 \le \mu \le 0.4)$. At the same time, the NMI values for all methods fall as the value rises. As a result, all of the approaches perform poorly in the range of $\mu \ge 0.7$.

However, we note that the DAC-HPP of performance decline is slower than that of other comparison techniques. Aside from $\mu = 0.5$, DAC-HPP also outperforms the competition, especially on networks with larger values that are more complex. In the network with $\mu = 0.5$, the CDE has the highest performance, NMI = 0.902, while DAC-HPP has the second-highest performance with NMI = 0.8892. Additionally, Fig. 4 shows that DAC-HPP outperforms competitors in terms of RI with a sizable difference. Based on the RI in Fig. 4, the CDE and SENet

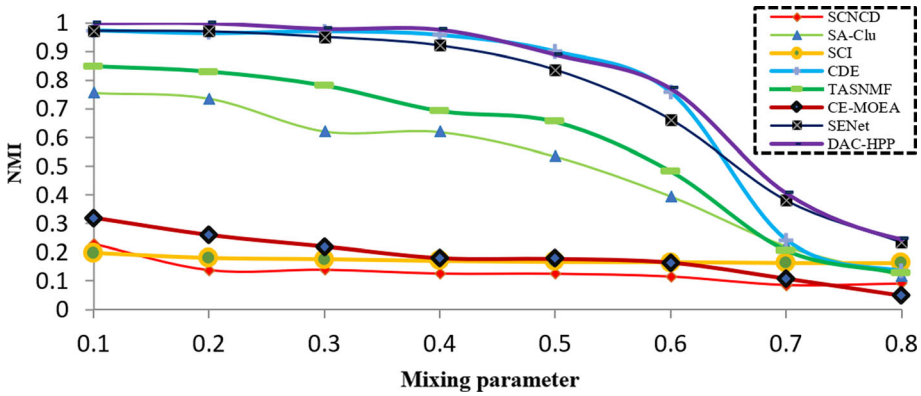

Fig. 3 Performance comparison of methods in terms of NMI on synthetic networks with $v = 0.5$ and $\mu \in [0.1, 0.8]$

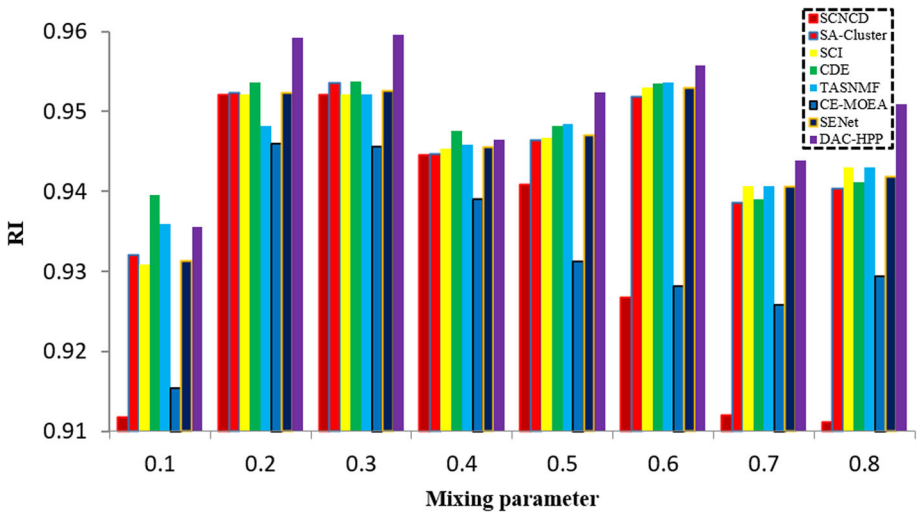

Fig. 4 Performance comparison of methods in terms of RI on synthetic networks with $v = 0.5$ and $\mu \in [0.1, 0.8]$

Table 5 Quality evaluation in terms of modularity on synthetic networks

Methods	μ							
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
SCNCD	0.1162	0.2833	0.2654	0.0191	0.1224	0.0148	0.0660	0.0435
SA-Cluster	0.5494	0.5005	0.3188	0.3375	0.2573	0.1940	0.1591	0.1455
SCI	0.2957	0.1850	0.1757	0.0747	0.0497	0.0378	0.0310	0.0136
CDE	0.8059	0.7315	0.6355	0.5460	0.4246	**0.3309**	0.1651	0.1554
TASNMF	0.4370	0.4223	0.3575	0.2875	0.2167	0.1466	0.1023	0.0429
CE-MOEA	0.1340	0.1280	0.1331	0.1340	0.1259	0.0964	0.0361	0.0334
SENet	0.8099	0.7244	0.6188	0.5250	0.3958	0.2729	0.1408	0.1025
DAC-HPP	**0.8106**	**0.7409**	**0.6276**	**0.5499**	**0.4288**	0.3093	**0.1655**	**0.1406**

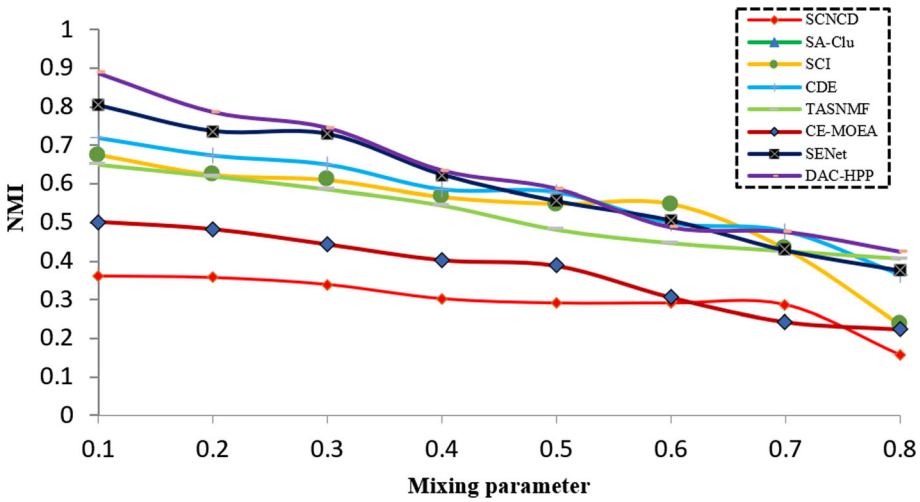

Fig. 5 Performance comparison of methods in terms of NMI on synthetic networks with $v \in [0.1, 0.8]$ $\mu = 0.5$

methods are ranked second and third, respectively. Additionally, when compared to other methods, these methods have an average rank of 2 and 3 according to the RI shown in Fig. 4.

In this experiment, the quality-based metric, modularity, is used to evaluate the quality of discovered communities by comparison methods, and its numerical results are reported in Table 5. It is seen that DAC-HPP has the best or second-best performance in all cases. In other words, DAC-HPP works better in 7 cases, while CDE has better performance in the other two. This experiment shows the power of DAC-HPP to detect communities of high quality.

In every case, DAC-HPP is shown to have the best or second-best performance. In other words, DAC-HPP performs better in 6 circumstances, whereas CDE performs better in two different situations. This experiment demonstrates the DAC-HPP's ability to detect high-quality clusters compared to other methods.

As was already established, most community detection methods cannot accurately identify the community structure for $\mu \geq 0.7$. Additionally, DAC-HPP performs second best in the $\mu = 0.5$ for NMI and modularity. As a result, in the second experiment, the networks with $\mu = 0.5$ and $\mu = 0.6$ faced competition from the comparison methods. This experiment also aims to determine how sensitive community detection algorithms are to v, representing the attribute noise value. In order to achieve these goals, we created various networks in the second experiment using the parameters $N = 1000$, $k = 25$, $max - k = 40$, $min - c = 60$, $max - c = 100$, $nattr = 10$ (int $attr$ [1, 15]), $v = 0.5$, and the mixing parameters [0.1, 0.8].

We compare these networks' performance in terms of NMI, RI, and modularity using the comparison techniques. The NMI outcomes of this experiment on the networks with the two mixing parameter values μ μ= 0.5 and $\mu = 0.6$ are shown in Figs. 5 and 6. These findings demonstrate that DAC-HPP consistently has the greatest NMI value. The second and third-best results, respectively, belong to SENet and CDE. The gradient of the charts for the two algorithms, SCNCD and CE-MOEA, are both extremely tiny, indicating that the v parameter has little impact on the two algorithms' performance. The SCNCD approach with an ideal alpha has the lowest NMI in both scenarios ($\mu = 0.5$ and μ=0.6).

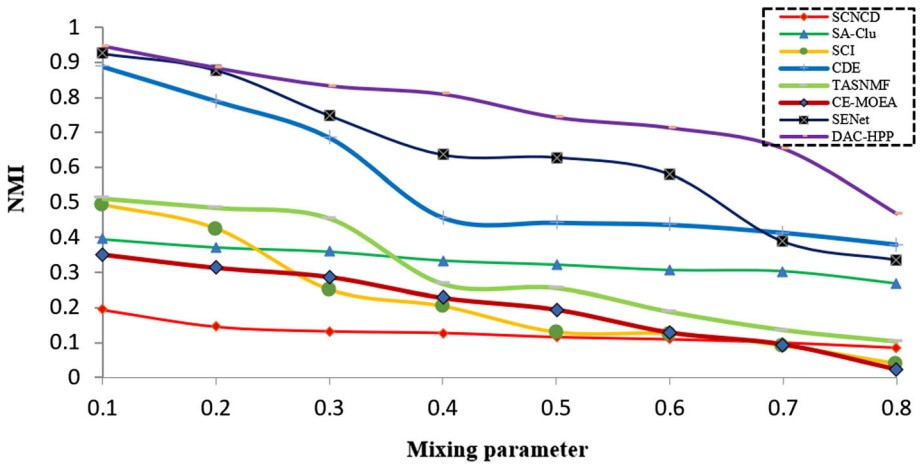

Fig. 6 Performance comparison of methods in terms of NMI on synthetic networks with $v \in [0.1, 0.8]$ $\mu = 0.6$

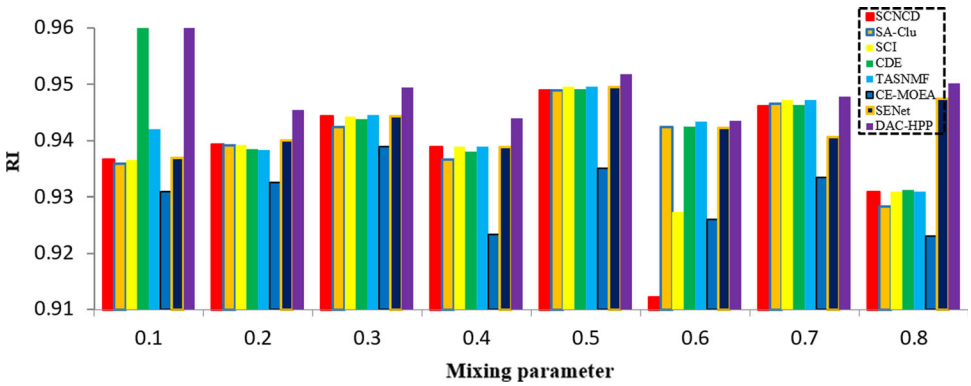

Fig. 7 Performance comparison of methods in terms of RI on synthetic networks with $v \in [0.1, 0.8]$ $\mu = 0.5$

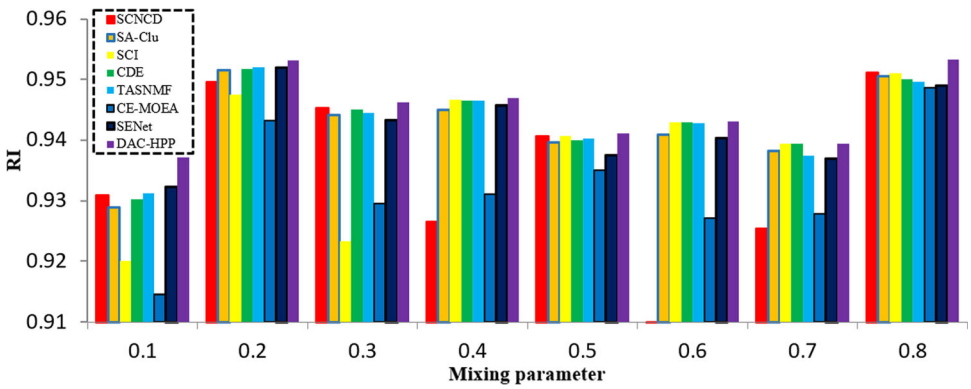

Fig. 8 Performance comparison of methods in terms of RI on synthetic networks with $v \in [0.1, 0.8]$ $\mu = 0.6$

Figures 7 and 8 depict the outcomes of method comparisons in terms of RI, demonstrating the superiority of DAC-HPP over the competition. These findings show that, following DAC-HPP, two approaches, SENet and CDE, perform the best when measured against other comparison methods.

Additionally, Table 6 shows the numerical results of methods in terms of modularity. Also, in this situation, DAC-HPP consistently performs at the top and second-highest levels. Apart from v = 0.1, which has the second-highest NMI value after SENet in this scenario, the DAC-HPP has the greatest NMI values overall when μ = 0.5. In other words, DAC-HPP outperforms all other comparison methods by raising the network's attribute noise. At various values of v, DAC-HPP has the highest and second-highest values in the μ = 0.6. The CDE method's first rank modularity is acceptable when attribute noise is μ = 0.3 or μ = 0.5. However, these methods become less effective as the v values rise. These test results show that DAC-HPP is

Table 6 Quality evaluation in terms of modularity on synthetic networks with $\mu \in \{0.5, 0.6\}$ and $v \in [0.1, 0.8]$

μ	Methods	v							
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
0.5	SCNCD	0.2170	0.1904	0.0909	0.0635	0.0539	0.0204	0.0157	0.0201
	SA-Clu	0.2627	0.2603	0.2566	0.2250	0.2006	0.2044	0.2010	0.1910
	SCI	0.2801	0.1880	0.1713	0.1112	0.08012	0.0704	0.0307	0.0152
	CDE	0.285	0.2774	0.2706	0.2699	0.2447	0.2378	0.2378	0.2178
	TASNMF	0.2081	0.1964	0.1449	0.1044	0.0805	0.0620	0.0587	0.0488
	CE-MOEA	0.1889	0.1719	0.1528	0.1454	0.1244	0.0983	0.0485	0.0309
	SENet	0.3582	0.3402	0.3067	0.2893	0.2827	**0.2769**	0.2508	0.1959
	DAC-HPP	**0.3921**	**0.3826**	**0.3710**	**0.3303**	**0.2989**	0.2709	**0.2654**	**0.2301**
0.6	SCNCD	0.1147	0.814	0.773	0.0628	0.0621	0.0490	0.0294	0.0209
	SA-Clu	0.1727	0.1681	0.1494	0.1433	0.1612	0.1556	0.1485	0.1440
	SCI	0.1967	0.1808	0.0866	0.0844	0.0399	0.0421	0.0269	0.0195
	CDE	0.2579	0.2526	**0.2451**	**0.2183**	0.2004	0.1694	0.1784	0.1589
	TASNMF	0.1054	0.0983	0.0656	0.0451	0.0250	0.0225	0.0136	0.0107
	CE-MOEA	0.1604	0.1407	0.126	0.0806	0.0707	0.0506	0.0350	0.0299
	SENet	0.1843	0.1678	0.1581	0.1405	0.1309	0.1219	0.1092	0.1048
	DAC-HPP	**0.2618**	**0.2536**	0.2081	0.1969	**0.2014**	**0.2009**	**0.1994**	**0.1884**

a better way to find high-quality communities than other methods.

4.6 Evaluation of real-world networks

In this subsection, we conducted a comprehensive comparison of DAC-HPP with other existing methods on real-world networks characterized by a high dimension of attributes. The performance evaluation was carried out on six diverse datasets: Texas, Cornell, Washington, Wisconsin, Cora, and PubMed. To assess the effectiveness of DAC-HPP, we employed three widely-used evaluation metrics: NMI, modularity, and RI.

The results presented in Table 7 provide valuable insights into the performance of DAC-HPP across the different datasets. Starting with the Texas dataset, DAC-HPP demonstrated outstanding performance by achieving the highest scores in all three metrics—NMI, modularity, and RI. This exceptional performance indicates that DAC-HPP effectively captures the underlying patterns and structures within the dataset, leading to highly accurate and meaningful clustering results. Similarly, in the Cornell dataset, DAC-HPP emerged as the top performer, surpassing other methods in all three metrics. This showcases the robustness and reliability of DAC-HPP in capturing the intrinsic characteristics of the dataset and generating high-quality clusters. Moving on to the Washington dataset, DAC-HPP ranked second in NMI, but it outperformed other methods in terms of modularity and RI. This indicates that DAC-HPP successfully identifies cohesive and well-connected communities within the network, resulting in superior modularity and RI scores.

In the Wisconsin dataset, DAC-HPP once again demonstrated its superiority by securing the first position in all three metrics. This highlights its ability to effectively handle high-dimensional attribute data and produce accurate clustering results. Analyzing the Cora dataset, DAC-HPP showcased its strength by securing the first position in both NMI and modularity. This indicates its capability to capture meaningful patterns and uncover cohesive communities within the dataset. Although DAC-HPP did not achieve the highest RI score in the Cora dataset, its strong performance in the other two metrics underscores its effectiveness in community discovery tasks. Lastly, in the PubMed dataset, DAC-HPP achieved the highest scores in modularity and RI. This highlights its ability to effectively identify well-connected and cohesive communities within the network, leading to a higher modularity score. Additionally, the high RI score reflects DAC-HPP's success in accurately clustering node pairs. Overall, the evaluation results consistently demonstrate that DAC-HPP outperforms other comparison methods across the NMI, modularity, and RI metrics. This highlights the effectiveness of DAC-HPP in accurately uncovering community structures and discovering meaningful patterns in real-world networks characterized by high-dimensional attributes.

Table 7 Quality evaluation in terms of NMI, Modularity, and RI on real-world networks

Data-set	Method	NMI ↑	Modularity ↑	RI ↑
Texas	SCNCD	0.1646	0.0112	0.6302
	SA-Cluster	0.0459	0.0336	0.6302
	SCI	0.2153	0.0940	0.6534
	CDE	0.3208	0.3151	0.6367
	TASNMF	0.3142	0.0124	0.6434
	CE-MOEA	0.1084	0.2054	0.6237
	SENet	0.1481	0.1145	0.6302
	DAC-HPP	**0.3284**	**0.3630**	**0.7069**
Cornell	SCNCD	0.0938	0.1797	0.7176
	SA-Cluster	0.0457	0.0282	0.6111
	SCI	0.1516	0.0496	0.7190
	CDE	0.3403	0.1383	0.7193
	TASNMF	0.1525	0.0337	0.7183
	CE-MOEA	0.1329	0.1095	0.5099
	SENet	0.0265	0.1068	0.7181
	DAC-HPP	**0.3505**	**0.4076**	**0.7233**
Washington	SCNCD	0.1111	0.0162	0.5827
	SA-Cluster	0.0546	0.0226	0.4620
	SCI	0.1304	0.1063	0.5144
	CDE	**0.4079**	0.3625	0.4660
	TASNMF	0.1999	0.1005	0.4521
	CE-MOEA	0.1699	0.1720	0.4896
	SENet	0.0935	0.1134	0.5048
	DAC-HPP	0.4004	**0.3889**	**0.7233**
Wisconsin	SCNCD	0.0461	0.1459	0.6751
	SA-Cluster	0.0294	0.0441	0.3475
	SCI	0.1823	0.0916	0.6336
	TASNMF	0.1848	0.1641	0.6792
	CE-MOEA	0.1453	0.1587	0.6044
	SENet	0.0420	0.1217	0.6751
	DAC-HPP	**0.4401**	**0.3809**	**0.6802**
Cora	SCNCD	0.1125	0.1445	0.7603
	SA-Cluster	0.1190	0.2835	0.7535
	SCI	0.2138	0.2357	0.8042
	CDE	0.5037	0.3703	0.8118
	TASNMF	0.1225	0.1732	0.7942
	CE-MOEA	0.2503	0.0358	0.7535
	SENet	0.3469	0.3305	**0.8134**
	DAC-HPP	**0.5430**	**0.4282**	0.8119
PubMed	SCNCD	0.0964	0.1834	0.5273
	SA-Cluster	0.1005	0.3127	0.5423
	SCI	0.1415	0.3102	0.5617
	CDE	0.4032	**0.4756**	0.7121
	TASNMF	0.1848	0.2203	0.6508
	CE-MOEA	–	–	–
	SENet	0.1143	0.3106	0.5624
	DAC-HPP	**0.4099**	0.4631	**0.7356**

5 Discussion

Our proposed method, Deep Attributed Clustering with High-order Proximity Preserve (DAC-HPP), introduces several significant advancements in the field of attributed graph clustering. Through rigorous comparisons with existing techniques, we have demonstrated the superior performance and accuracy of DAC-HPP in terms of clustering effectiveness. By employing an end-to-end framework, DAC-HPP optimizes both embedding learning and node clustering jointly, resulting in improved outcomes compared to separate approaches. The construction of a consensus matrix, which integrates diverse degrees of proximity, facilitates the fusion of structural and attribute information, capturing the global characteristics of the network. The insights derived from the consensus matrix representation have revealed novel findings and patterns in attributed graph clustering. Moreover, DAC-HPP exhibits practical applicability in various domains, including social network analysis, recommendation systems, and bioinformatics. Nevertheless, we acknowledge the existing challenges, such as scalability, computational efficiency, and applicability to large-scale networks, that need to be addressed. Future research directions could explore the incorporation of temporal dynamics or the integration of additional data modalities. In conclusion, DAC-HPP significantly contributes to the field of attributed graph clustering by refining existing techniques and providing a deeper understanding of the underlying processes.

6 Conclusion

Attributed graph clustering plays an important role in complex network analysis. It provides us with an understanding of the structures and functions of a complex network. In this work, we propose a new attributed graph clustering method based on deep clustering methods called Deep Attributed Clustering with High-order Proximity Preserve, or DAC-HPP, which is a deep end-to-end clustering in attributed networks with high structural cohesiveness and attribute homogeneity. By merging the different degrees of proximity, we construct a consensus matrix representation of the network that serves as the basis for deep clustering. By using a modified Random Walk with Restart on high-order proximities, it is feasible to capture the k-order of structure and the attributed information. Extensive experiments were conducted on synthetic and real-world benchmark datasets to demonstrate the validity of our model, and the results show that our model achieves new state-of-the-art performance. In recent years, detecting communities in multi-modal networks

based on their higher-order structure has drawn increasing attention. However, capturing the higher-order relations between nodes is much more difficult in different modals because the topological structure varies. Therefore, how to combine the higher-order structural information from all modals of the network to discover the community structure is more challenging than in a single-modal network. We are considering this topic for our future work.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions. The authors have not disclosed any funding.

Data availability The data that support the findings of this study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

1. Newman ME (2003) The structure and function of complex networks. SIAM Rev 45(2):167–256
2. Su X, Xue S, Liu F, Wu J, Yang J, Zhou C, Hu W, Paris C, Nepal S, Jin D, et al (2022) A comprehensive survey on community detection with deep learning. IEEE Trans Neural Netw Learn Syst
3. Sieranoja S, Fränti P (2022) Adapting k-means for graph clustering. Knowl Inf Syst 64(1):115–142
4. Berahmand K, Bouyer A, Vasighi M (2018) Community detection in complex networks by detecting and expanding core nodes through extended local similarity of nodes. IEEE Trans Comput Soc Syst 5(4):1021–1033
5. Chunaev P (2020) Community detection in node-attributed social networks: a survey. Comput Sci Rev 37:100286
6. Liu L, Chen P, Luo G, Kang Z, Luo Y, Han S (2022) Scalable multi-view clustering with graph filtering. Neural Comput Appl 1–9
7. Wang C, Pan S, Celina PY, Hu R, Long G, Zhang C (2022) Deep neighbor-aware embedding for node clustering in attributed graphs. Pattern Recogn 122:108230
8. Berahmand K, Haghani S, Rostami M, Li Y (2020) A new attributed graph clustering by using label propagation in complex networks. J King Saud Univ Comput Inf Sci
9. Xu W, Liu X, Gong Y (2003) Document clustering based on non-negative matrix factorization. In: Proceedings of the 26th annual international ACM SIGIR conference on research and development in information retrieval, pp. 267–273
10. He C, Fei X, Cheng Q, Li H, Hu Z, Tang Y (2021) A survey of community detection in complex networks using nonnegative matrix factorization. IEEE Trans Comput Soc Syst
11. Golzari Oskouei A, Balafar MA, Motamed C (2022) Edcwrn: efficient deep clustering with the weight of representations and the help of neighbors. Appl Intell 1–23
12. Chen M-S, Lin J-Q, Li X-L, Liu B-Y, Wang C-D, Huang D, Lai J-H (2022) Representation learning in multi-view clustering: a literature review. Data Sci Eng 1–17
13. Min E, Guo X, Liu Q, Zhang G, Cui J, Long J (2018) A survey of clustering with deep learning: from the perspective of network architecture. IEEE Access 6:39501–39514
14. Wang C, Pan S, Celina PY, Hu R, Long G, Zhang C (2022) Deep neighbor-aware embedding for node clustering in attributed graphs. Pattern Recogn 122:108230
15. Xu H, Xia W, Gao Q, Han J, Gao X (2021) Graph embedding clustering: graph attention auto-encoder with cluster-specificity distribution. Neural Netw 142:221–230
16. Bothorel C, Cruz JD, Magnani M, Micenkova B (2015) Clustering attributed graphs: models, measures and methods. Netw Sci 3(3):408–444
17. Rostami M, Oussalah M, Berahmand K, Farrahi V (2023) Community detection algorithms in healthcare applications: a systematic review. IEEE Access
18. Zhou Y, Cheng H, Yu JX (2009) Graph clustering based on structural/attribute similarities. Proc VLDB Endow 2(1):718–729
19. Meng F, Rui X, Wang Z, Xing Y, Cao L (2018) Coupled node similarity learning for community detection in attributed networks. Entropy 20(6):471
20. Jia C, Li Y, Carson MB, Wang X, Yu J (2017) Node attribute-enhanced community detection in complex networks. Sci Rep 7(1):1–15
21. Li Y, Jia C, Kong X, Yang L, Yu J (2017) Locally weighted fusion of structural and attribute information in graph clustering. IEEE Trans Cybern 49(1):247–260
22. Alinezhad E, Teimourpour B, Sepehri MM, Kargari M (2020) Community detection in attributed networks considering both structural and attribute similarities: two mathematical programming approaches. Neural Comput Appl 32(8):3203–3220
23. Zhou Y, Cheng H, Yu JX (2009) Graph clustering based on structural/attribute similarities. Proc VLDB Endow 2(1):718–729
24. Lee DD, Seung HS (1999) Learning the parts of objects by non-negative matrix factorization. Nature 401(6755):788–791
25. Li Y, Sha C, Huang X, Zhang Y (2018) Community detection in attributed graphs: an embedding approach. In: Proceedings of the AAAI conference on artificial intelligence, vol. 32
26. Wang X, Jin D, Cao X, Yang L, Zhang W (2016) Semantic community identification in large attribute networks. In: Proceedings of the AAAI conference on artificial intelligence, vol. 30
27. Lu D-D, Qi J, Yan J, Zhang Z-Y (2022) Community detection combining topology and attribute information. Knowl Inf Syst 64(2):537–558
28. Sun J, Zheng W, Zhang Q, Xu Z (2021) Graph neural network encoding for community detection in attribute networks. IEEE Trans Cybern 52(8):7791–7804
29. He C, Zheng Y, Fei X, Li H, Hu Z, Tang Y (2021) Boosting nonnegative matrix factorization based community detection with graph attention auto-encoder. IEEE Trans Big Data

30. Lafabregue B, Weber J, Gançarski P, Forestier G (2022) End-to-end deep representation learning for time series clustering: a comparative study. Data Min Knowl Disc 36(1):29–81
31. Jin D, Yu Z, Jiao P, Pan S, He D, Wu J, Yu P, Zhang W (2021) A survey of community detection approaches: from statistical modeling to deep learning. IEEE Trans Knowl Data Eng
32. Su X, Xue S, Liu F, Wu J, Yang J, Zhou C, Hu W, Paris C, Nepal S, Jin D, et al (2022) A comprehensive survey on community detection with deep learning. IEEE Trans Neural Netw Learn Syst
33. Pan S, Hu R, Fung S-F, Long G, Jiang J, Zhang C (2019) Learning graph embedding with adversarial training methods. IEEE Trans Cybern 50(6):2475–2487
34. Zhang Z, Yang H, Bu J, Zhou S, Yu P, Zhang J, Ester M, Wang C (2018) Anrl: attributed network representation learning via deep neural networks. In: Ijcai 18:3155–3161
35. Hong R, He Y, Wu L, Ge Y, Wu X (2019) Deep attributed network embedding by preserving structure and attribute information. IEEE Trans Syst Man Cybern Syst 51(3):1434–1445
36. Xu H, Xia W, Gao Q, Han J, Gao X (2021) Graph embedding clustering: graph attention auto-encoder with cluster-specificity distribution. Neural Netw 142:221–230
37. Wang C, Pan S, Celina PY, Hu R, Long G, Zhang C (2022) Deep neighbor-aware embedding for node clustering in attributed graphs. Pattern Recogn 122:108230
38. Zhang X, Liu H, Wu X-M, Zhang X, Liu X (2021) Spectral embedding network for attributed graph clustering. Neural Netw 142:388–396
39. Cai H, Zheng VW, Chang KC-C (2018) A comprehensive survey of graph embedding: problems, techniques, and applications. IEEE Trans Knowl Data Eng 30(9):1616–1637
40. Pan J-Y, Yang H-J, Faloutsos C, Duygulu P (2004) Automatic multimedia cross-modal correlation discovery. In: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining, pp. 653–658
41. Jin W, Jung J, Kang U (2019) Supervised and extended restart in random walks for ranking and link prediction in networks. PLoS ONE 14(3):0213857
42. Xia F, Liu J, Nie H, Fu Y, Wan L, Kong X (2019) Random walks: a review of algorithms and applications. IEEE Trans Emerging Top Comput Intell 4(2):95–107
43. Zhu D, Cui P, Zhang Z, Pei J, Zhu W (2018) High-order proximity preserved embedding for dynamic networks. IEEE Trans Knowl Data Eng 30(11):2134–2144
44. Saebi M, Ciampaglia GL, Kaplan LM, Chawla NV (2020) Honem: learning embedding for higher order networks. Big Data 8(4):255–269
45. Sun X, Yu Y, Liang Y, Dong J, Plant C, Böhm C (2021) Fusing attributed and topological global-relations for network embedding. Inf Sci 558:76–90
46. Socher R, Pennington J, Huang EH, Ng AY, Manning CD (2011) Semi-supervised recursive autoencoders for predicting sentiment distributions. In: Proceedings of the 2011 conference on empirical methods in natural language processing, pp. 151–161
47. Bo D, Wang X, Shi C, Zhu M, Lu E, Cui P (2020) Structural deep clustering network. In: Proceedings of the web conference, pp. 1400–1410
48. Xie J, Girshick R, Farhadi A (2016) Unsupervised deep embedding for clustering analysis. In: International conference on machine learning, pp. 478–487. PMLR
49. Li W, Jiang S, Jin Q (2018) Overlap community detection using spectral algorithm based on node convergence degree. Futur Gener Comput Syst 79:408–416
50. Zhou Y, Cheng H, Yu JX (2010) Clustering large attributed graphs: an efficient incremental approach. In: 2010 IEEE International conference on data mining, pp. 689–698. IEEE
51. Elhadi H, Agam G (2013) Structure and attributes community detection: comparative analysis of composite, ensemble and selection methods. In: Proceedings of the 7th workshop on social network mining and analysis, pp. 1–7
52. Lancichinetti A, Fortunato S, Radicchi F (2008) Benchmark graphs for testing community detection algorithms. Phys Rev E 78(4): 046110
53. Sen P, Namata G, Bilgic M, Getoor L, Galligher B, Eliassi-Rad T (2008) Collective classification in network data. AI Mag 29(3):93–93
54. Strehl A, Ghosh J (2002) Cluster ensembles-a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3:583–617
55. Rand WM (1971) Objective criteria for the evaluation of clustering methods. J Am Stat Assoc 66(336):846–850
56. Chen M, Kuzmin K, Szymanski BK (2014) Community detection via maximization of modularity and its variants. IEEE Trans Comput Soc Syst 1(1):46–65