Information Sciences 657 (2024) 119969

Contents lists available at ScienceDirect

Information Sciences

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

Detecting communities in attributed networks through bi-direction penalized clustering and its application

Hu Yang [a], Wenjing Xiang [a], Jar-Der Luo [b], Qiuyan Zhang [c,*]

[a] School of Information, Central University of Finance and Economics, Beijing, China
[b] Department of Sociology, Tsinghua University, Beijing, China
[c] School of Statistics, Capital University of Economics and Business, Beijing, China

ARTICLE INFO

Keywords:
Data science
Community detection
Representation learning
Embedding dimension
Penalized clustering

ABSTRACT

Exploiting heterogeneous information in attributed networks to improve the performance of community detection has attracted considerable research attention. Although variational graph autoencoder (VGAE)-based methods have been proven to be effective strategies, they perform community detection based on assumptions regarding the dimension of embedding and the number of communities, limiting their effectiveness and applicability. In this study, we combined VGAE-based methods and a bi-direction penalized clustering algorithm (BiPClust) for community detection. Our approach addresses the issues of dimension selection and community number determination by automatically optimizing penalized clustering. Both the computational algorithm and statistical theorems confirm that BiPClust effectively mitigates the impacts of redundant embedding and determines the unknown number of communities. Furthermore, applying the proposed methods to community detection on benchmark datasets and syndicated investment networks in China reveals that BiPClust surpasses other methods in performance.

1. Introduction

Communities exist in various forms worldwide and have been investigated since as early as the 1920s in the fields of sociology and social anthropology [30]. Their prevalence extends across various systems, including social structures, information networks, ecosystems, and even syndicated investment networks [1]. Exploring communities among networks helps us understand the structure or organization within these interconnected systems, shedding light on the patterns of relationships and interactions among their components. For example, in the venture capital (VC) market, venture capitalists (VCs) consistently engage in syndicated investments with other VCs to mitigate investment risks and compensate for shortcomings [36]. The choice of VC investment partners is not arbitrary; instead, it is driven by preferences, leading to the formation of the phenomenon known as the venture capital community [17]. Hence, uncovering the community structure in the co-investment network provides valuable insights into the overall functionality of the system. This involves understanding why VCs are more inclined to syndicate with a small group of preferred partners rather than external ones.

Community detection encompasses partitioning social actors based on their features and connections into densely knitted, highly related groups that are well-separated from each other [4]. In the past ten years, researchers in computer science have extensively studied community detection by leveraging network topological structures [19,40,3] and semantic information [4]. Many traditional

* Corresponding author.
E-mail address: qyzhang@cueb.edu.cn (Q. Zhang).

https://doi.org/10.1016/j.ins.2023.119969
Received 3 January 2023; Received in revised form 26 November 2023; Accepted 27 November 2023

Available online 30 November 2023
0020-0255/

techniques, such as graph-based approaches [3,38], and distance-based clustering [47,22], have proposed learning communities solely from their topology structure, neglecting the attribute information of the nodes; this has led to misunderstandings of the internal characteristics of the nodes and the mechanisms behind the formation of the network structure [46]. For example, in a protein-protein interaction network [49], the interactions between different proteins form a network, with each node representing a protein and its structure serving as the attribute. In such cases, using both structural and attributed information is believed to yield more qualitative community detection results and lend meaning to the identified communities. However, integrating both the topological structure and the attribute information for community detection remains a challenging task [4], falling under the domain of the attributed network or graph learning. A natural strategy for attributed network learning is to initially derive embedding from the attributed network and subsequently apply unsupervised learning techniques for community detection. For instance, several studies employ nonnegative matrix factorization methods to achieve fusion of topological and attributed information of networks, followed by the utilization of k-means or spectral clustering to discover cliques [14]. Thus, the task of community detection for attributed networks can be reframed as obtaining the attributed network embedding and performing clustering.

Inspired by the remarkable achievements of deep learning (DL) [32], several artificial neural network (ANN) algorithms have garnered attention and positioned themselves as strong contenders for statistical community detection methods owing to their superior prediction accuracy [12]. The essence of DL is to extract lower-dimensional vectors from high-dimensional data representing complex structural relationships [4,26,45], rendering it suitable for modeling and learning new representations of attributed networks. Therefore, it enables knowledge discovery via state-of-the-art machine learning and data mining techniques. The representation framework can additionally incorporate non-structural features, such as node attributes, to enhance the knowledge of community memberships [10,18]. Moreover, groups of information from nodes [26] or edges [31] can be jointly recognized with special attention in the DL process, leading to effective community detection results. Recently, state-of-the-art representation learning neural networks, such as graph convolutional encoders or graph attention encoders, have been developed [6]. As a generative technique, the variational graph auto-encoder (VGAE) utilizes two graph convolutional layers and employs graph structure reconstruction to learn the representation of attribute networks with high performance [20], establishing itself as a superior graph representation technique. Therefore, by combining clustering algorithms, VGAE, along with its various extensions such as C-VGAE, VGAER, VGAECD [27], modularity-aware GAE (MAGAE), modularity-aware VGAE (MAVGAE) [45], linear modularity-aware VGAE (LMAVGAE) and GCN-based modularity-aware VGAE (GMAVGAE) [12], has been demonstrated as effective strategies for community detection in attributed networks [4,50].

While many DL methods can fuse the topology and node attributes to learn embedding representations of attributed networks, automatically determining the dimension of the embedding remains an open question for most DL techniques. This aspect is crucial as it can significantly impact their performance. To address this challenge, the most straightforward approach involves setting the dimension of the embedding to a fixed number based on prior information or experience, but it may be limited if the actual dimensionality of the data differs from the assumed dimensionality. For example, some VGAE-based embedding methods [12] set the number of dimension to 16. An alternative approach involves conducting a grid search across a range of potential dimensions and assessing the clustering performance of the method on a validation set. In such cases, prior methodologies treated dimensionality as a hyperparameter, which could be determined through tuning techniques (such as cross-validation [44]) or selected based on individual experience, often without excessive over-parametrization risk (e.g., 100, 200, or 300 dimensions) [10]. Despite the introduction of a recent metric [11] intended to ascertain a suitable embedding dimension, we discovered that it cannot be directly applied to select the embedding dimension for attributed networks, as it exclusively considers networks without attributes. Another strategy is to assign a relatively large value to the dimension to retain as much informative data from attributed networks as possible. However, this may result in the inclusion of unnecessary embeddings due to the separation of embedding learning and downstream tasks. To mitigate the impact of these embeddings in the presence of high background noise, one approach is to conduct optimal subset selection [21] during downstream learning, such as Clustering Objects on Subsets of Attributes (COST) [9]. Similar to stepwise regression, these methods ignore the stochastic error inherent in the stages of feature selection, with a different subset of features leading to different clustering performances. Regularization techniques involve imposing certain prior distributions on model parameters to train simple and/or sparse models [23]; therefore, these techniques have been widely used in high-dimensional data analysis, particularly after the successful application of l_1-norm penalty (lasso) [34]. In addition, regularization techniques have been also extended to clustering algorithms, such as the sparse clustering algorithm [37] and regularized k-means clustering [33], to achieve stable results.

In addition to dimension selection, determining the number of communities is a challenging issue. Once the embeddings of attribute graphs are obtained, state-of-the-art unsupervised learning methods can be applied via clustering for community detection [7]. To solve the aforementioned issue, clustering validation is used to evaluate the quality of clustering results and obtain the optimal number of clusters (such as k-means and spectral clustering [7,15]). The clustering algorithm automatically divides observations into groups so that similar data objects are placed within one cluster, whereas dissimilar ones are assigned to different clusters, possibly separating noise. Most of them have focused on heuristically determining the number of clusters, such as optimizing the clustering validation, i.e., the process of estimating how well a partition fits the underlying data structure [35]. The clustering stability criterion [33,37], which measures the robustness of any given clustering algorithm, has been used for selecting the number of clusters using cross-validation. However, an inappropriate choice or application of the criterion can lead to unstable results. Therefore, the penalized regression-based clustering (PClust) method has been proposed for unsupervised learning without prior knowledge regarding the number of clusters [25]. PRclust obtains the optimal number of clusters when applied to data with few features but generates unstable results for less informative features.

To determine the embedding dimension and community number, we propose a novel framework for community detection based on a VGAE-based bi-direction penalized clustering algorithm (BiPClust). This framework first uses the VGAE-based method to obtain

graph representations by assigning a relatively large value to the dimension to keep as much informative data of attributed networks as possible. Then, a new clustering algorithm (bi-direction penalized clustering) is proposed to learn clusters based on the learned embeddings. This algorithm can automatically determine the number of clusters using fused lasso penalty and counteract the effects of less informative embeddings using group lasso penalty. Based on the theorem analysis, we demonstrate why the proposed model performs well in selecting informative variables. Further, we demonstrate asymptotic estimation in our proposed method with fused lasso penalty along with its selection consistency. The proposed clustering process is also applicable in case of diverging dimensions. We describe the computational algorithm by presenting the mechanism through which the proposed algorithm selects variables and determines the number of clusters via optimization. In addition, we prove the effectiveness of this method by employing it on two benchmark datasets and comparing its performance to those of other algorithms for the same datasets. Finally, we demonstrate the application of the method to the Chinese VC co-investment network to effectively identify the community structure of the network and analyze the final results.

The remainder of the paper is organized as follows: Section 2 provides an overview of the development of community detection using network embedding. Section 3 proposes the method we applied for community detection. Our data and experimental work are outlined in Section 4. Section 5 presents the results of real data analyses. We present the conclusions and discussion in Section 6.

2. Preliminaries

2.1. Problem settings

Given an attribute network as a three tuple $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$, where $\mathcal{V} = \{v_1, v_2, ..., v_n\} \in R^n$ is a set of actors, $\mathcal{E} \subset V \times \mathcal{V}$ is a set of relationships connecting two actors and $\mathcal{E} = \{e(v_i, v_j) : 1 \le i, j \le n\} \in R^{n \times n}$, where $e(v_i, v_j)$ can be simply denoted as e_{ij}, and $\mathcal{X} = \{X_1, X_2, ..., X_m\} \in R^{n \times m}$ is a set of attributes for the actors. Our goal is to partition the actors of the graph $\mathcal{G}$ into K clusters $C = \{C_1, C_2, ..., C_k\}$. For instance, a venture syndicated network $G \in \mathcal{G}$ is an attribute network, which has a set of venture capital firms $V \in \mathcal{V}$, a set of syndicated investment $E \in \mathcal{E}$, and also a set of features $X \in \mathcal{X}$, for each firm $x_i = \{x_{i1}, x_{i2}, ..., x_{im}\} \in R^m$, $i = 1, 2, ..., n$. Community detection of a syndicated investment network is achieved by dividing social actors with features and connections into densely knitted and highly related groups with each group well-separated from each other.

In the framework of deep learning, this process can typically be split into two tasks: embedding learning, which translates an attribute network into vectorized data directly usable by downstream tasks, and clustering, which divides actors into different groups based on the vectorizated data [1]. They are respectively defined as follows:

Definition 1. Embedding learning. It learns the representation from the attribute network G to obtain global information of both topology structure and node attributes simultaneously. Let $embedding(\cdot)$ be the learning algorithm that translates G to representation $Z \in R^{n \times d}$, which is

$$Z = embedding(G; \theta) \tag{1}$$

parameterized by θ, and the dimension of Z is d. The embedding learning is a transformer to map each node by capturing the structural and attribute information to a vector. In practice, embedding learning is an unsupervised learning task and the dimension of Z is considered as a hyperparameter.

Definition 2. Clustering. Given the representation Z of an attribute network and the number of clusters K, the task of community detection concerns dividing a set of actors into K groups $C = \{C_1, C_2, ..., C_K\}$, which places actors with high similarity in the same group and actors with low similarity in different groups. For instance, in k-means, the within-cluster sum of squares (WCSS) can be used as a criterion to detect clusters. The k-means cost function is given by

$$\arg\min \sum_{k=1}^{K} \sum_{i \in C_k} \left\{ dist(Z_{i\cdot} - \mu_{k\cdot}) \right\} \tag{2}$$

where $\mu_{k\cdot} \in \mu$ is the centroid of the actors in the kth cluster and $dist(Z_{i\cdot}, \mu_{k\cdot})$ is the dissimilarity measurement between $Z_{i\cdot}$ and $\mu_{k\cdot}$. If and only if $Z_{i\cdot}$ is close to $\mu_{k\cdot}$, it is assigned to the kth cluster to minimize the criteria.

2.2. Graph convolutional neural network (GCN)

Given an attribute network $G \in \mathcal{G}$ for n actors with m features, a weighted adjacency matrix A can be determined by G. We define the normalized Laplacian matrix is $L = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$ where D is the diagonal degree matrix with $d_{ii} = \sum_j a_{ij}$, a_{ij} is the ijth entry of A and I is the identify matrix. Thus, the eigenvectors matrix U and diagonal eigenvalues matrix Λ of L can be obtained directly. Based on the spectral graph analysis [5], we denote the spectral graph filter function as $\phi_\theta(\cdot)$ parameterized by θ in the Fourier domain and the spectral graph convolution operation as $\phi_\theta \star x = U \phi_\theta(\Lambda) U^T x$ which is used to implement convolution operation on signal x in the Fourier domain, where $\hat{x} = U^T x$ is the graph Fourier transform of a signal x and its inverse as $x = U\hat{x}$ [44], and $U \phi_\theta(\Lambda) U^T$ is the so-called graph convolution kernel. Based on the graph convolutional operation, a two-layer GCN can be defined by stacking it, which is given by

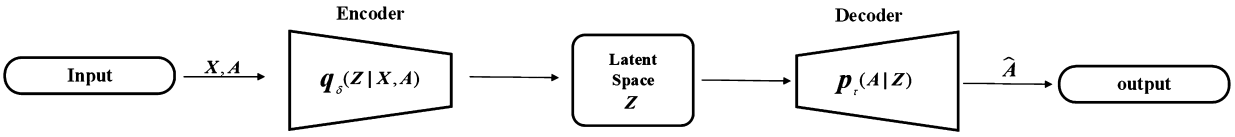

Fig. 1. Structure of variational graph auto-encoder.

$$Z = GCN(G) = \sigma(U\phi_{\theta_2}(\Lambda)U^T\sigma(U\phi_{\theta_1}(\Lambda)U^T X)) \tag{3}$$

parameterized by θ_1 and θ_2, $\sigma(\cdot)$ is the activation function (i.e. sigmoid function) and Z is the representation of G. To simplify the study, a linear model is applied to approximate $\phi_\theta \star x$, which is given by $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$, where $\tilde{A} = A + I$, and an entry of $\tilde{D}$ is $\tilde{d}_{ii} = \sum \tilde{a}_{ij}$ where $\tilde{a}_{ij}$ is an entry of $\tilde{A}$. With the approximation, we reformulate (3) to

$$Z = GCN(G) = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}\sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}XW_1)W_2) \tag{4}$$

where W_1 and W_2 are the learnable parameters for each layer. A two-layers GCN can be viewed as an encoder that transforms G to Z, in such a way that the downstream task can use the representation directly. In practice, GCN is sometimes a component of a deep learning framework so that unknown parameters can be estimated. Hence, the representation Z depends on the downstream task.

2.3. Variational graph auto-encoder (VGAE) and its extension

Similar to clustering, community detection is an unsupervised learning task in which no label information is available. To take the features of actors into consideration, a possible strategy is encoding the network with topology and attribute information into a vector space representation first, and then applying the classic unsupervised learning algorithm for community detection. Compared with other node embedding methods, VGAE-based method is a robust and accurate encoder for representation learning that fuses both the attributes and the connections of each actor. Besides, to learn the representation Z depending on G, a decoder needs to be connected to the encoder so that we can learn the unknown parameters of encoder. A simple case is VGAE, as shown in Fig. 1.

The first component of VGAE is an encoder of an attribute network in a d-dimensional real vector space or the so-called latent space. For example, a two-layer GCN can be used as an encoder transforming G to Z, where Z components are often called latent variables, since they are hidden from us. The second component is a decoder of latent variables Z to A or reconstructing A using Z. The decoder $p_\tau(G|Z)$ parameterized by τ takes the latent variables Z as an input to reconstruct G. Hence, the VGAE can be regarded as a stacked neural network, simply expressed as

$$\begin{aligned} Z &= GCN_W(G) \\ G &= p_\tau(G|Z) \end{aligned} \tag{5}$$

To estimate the unknown parameters $\{W, \tau\}$ in (5), we need to define the likelihood function of $p(G)$ through the generative model. Consider a statistical model that generates two random variables $Z \in \mathcal{Z}$ and $G \in \mathcal{G}$. However, we only get to see realizations of the G components and not the Z components. More specifically, although the unknown parameter $\{W^\star, \tau^\star\}$ generates pairs of samples of Z and G, only G can be observed. Following with VGAE [20], we suppose Z obeys the normal distribution $p(Z)$, the joint distribution of G and Z can be written as

$$p(G, Z) = p(Z)p_\tau(G|Z) \tag{6}$$

With the assumption that samples are collected independently from the normal distribution, $p(Z) = p(z_1, z_2, ..., z_n) = \prod_{i=1}^n p(z_i)$ and $z_i \sim N(0,1)$ for $i = 1, 2, ..., n$. Thus, the log-likelihood of G can be written as

$$\begin{aligned} log(p_\tau(G)) &= \int q_\delta(Z|G) log p_\tau(G) dZ \\ &= \int q_\delta(Z|G) log \frac{p_\tau(G,Z)}{p_\tau(Z|G)} dZ \\ &= \int q_\delta(Z|G) log(\frac{p_\tau(G,Z)}{q_\delta(Z|G)} \cdot \frac{q_\delta(Z|G)}{p_\tau(Z|G)}) dZ \\ &= \int q_\delta(Z|G) log \frac{p_\tau(G,Z)}{q_\delta(Z|G)} dZ + \int q_\delta(Z|G) log \frac{q_\delta(Z|G)}{p_\tau(Z|G)} dZ \\ &= \ell(p_\tau, q_\delta) + D_{KL}(q_\delta, p_\tau) \end{aligned} \tag{7}$$

where $q_\delta(Z|G)$ parameterized by δ is the variational distribution of Z conditioning on G, which models the encoder network for approximate posterior inference, and $q_\delta(Z|G) = \prod_{i=1}^n q_\delta(z_i|G)$ is based on the assumption that samples are independent. To estimate parameters in (7), we assume that the distribution of $q_\delta(z_i|G) = N(\mu_i(G), \sigma_i^2(G))$ where $\mu_i(\cdot)$ and $\sigma_i^2(\cdot)$ are learned using GCN in (4) and samples of $q_\delta(Z|G)$ are obtained from mean and variance using the reparameterization trick. In order to optimize the maximum log-likelihood in (7), Jensen's Inequality is used to get evidence lower bound (ELBO) of (7) which is given by,

$$
\begin{aligned}
log(p(G)) &\geq \ell(p_\tau, q_\delta) \\
&= \int q_\delta(Z|G) log \frac{p_\tau(G,Z)}{q_\delta(Z|G)} dZ \\
&= \int q_\delta(Z|G) log \frac{p_\tau(G|Z)p(z)}{q_\delta(Z|G)} dZ \\
&= \int q_\delta(Z|G) log \frac{p(z)}{q_\delta(Z|G)} dZ + \int q_\delta(Z|G) log p_\tau(G|Z) dZ \\
&= -D_{KL}(q_\delta, p) + E_{Z\sim q_\delta(Z|G)}[log(p_\tau(G|Z)))]
\end{aligned} \tag{8}
$$

where D_{KL} denotes the Kullback-Leibler (KL) divergence.

In practice, given G, the representation Z can be determined by $q_\delta(z_i|G)$ through a two-layer GCN (an encoder). Meanwhile, the reconstructed adjacency matrix $\hat{A}$ is obtained by Z, and $p_\tau(G|Z)$ is given by

$$
p_\tau(G|Z) = \prod_{(i,j)=(1,1)}^{(n,n)} p(A_{ij}|z_i, z_j) \tag{9}
$$

with the definition that $p(A_{ij} = 1|z_i, z_j) = \sigma(z_i z_j^T)$.

Inspired by the success of VGAE, some extensive approaches also have been developed to exploit the heterogeneous information in attributed networks recently. Besides VGAE, other examples improve encoder and decoder to adapt certain learning task and enhance its performance, such as Linear Graph AE (LGAE) which replace the GCN encoder by a simple linear model and Linear Modularity-Aware VGAE (LMAVGAE) and GCN-based Modularity-Aware VGAE (GMAVGAE) [12] which introduce a community-preserving message passing scheme to consider both the initial graph structure and modularity-based prior communities when computing embedding spaces. These extensive methods have similar working mechanism of VGAE. Although they perform well on attributed networks embedding, how to choose the dimension of latent embedding is still a challenge need to be puzzled.

2.4. K-means clustering

Community detection in network analysis is also called graph or network clustering [8]. Its goal is to identify high-quality groups whose vertices with higher similarity are in the same community rather than in different groups [1]. Although many algorithms for network analysis can utilize the structural data of network topology [16] for community detection, it is believed that using both structural and attribute information could yield more qualitative community detection results and provide sense to the detected communities. After translating network topology and node attributes to embedding (the vector structure data), some state-of-art unsupervised learning methods can be used to implement community detection. For instance, the k-means algorithm [8] aims to partition observations into different clusters, such that similar data objects remain within one cluster, whereas dissimilar ones are assigned to different clusters, possibly separating noise, serving as a prototype of the cluster [22].

Given data in matrix form $X = (X_{1\cdot}, X_{2\cdot}, ..., X_{n\cdot})^T$ that has n observations ($X_{i\cdot} = (x_{i1}, x_{i2}, ..., x_{im})^T$ is the ith observation). Assume that observations are divided into K clusters, and C_k is a set of observations in the kth cluster containing n_k observations. The corresponding clustering centroid of the kth cluster is $\mu = (\mu_{1\cdot}, \mu_{2\cdot}, ..., \mu_{K\cdot})^T$ for $k = 1, 2, ..., K$. Let $dist(X_{i\cdot}, X_{j\cdot}) \in [0, +\infty)$ be the distance between any two points $X_{i\cdot}$ and $X_{j\cdot}$ for $i, j = 1, 2, ..., n$. We introduce the total sum of distance $\sum_{k=1}^{K} \sum_{i\in C_k} dist(X_{i\cdot}, \mu_{k\cdot})$ as a criterion to divide observations into K clusters. Given the number of clusters K, an optimal centroid set $\mu^\star = \{\mu_1^\star, \mu_2^\star, ..., \mu_K^\star\}$ that minimizes the criterion can be obtained. A different number of clusters K yields different clustering centroids. Hence, to achieve optimal cluster validation, it is necessary to try many different K values to obtain the global optimal solution. The formalization of the k-means algorithm is defined as

$$
\arg\min \sum_{k=1}^{K} \sum_{i\in C_k} dist(X_{i\cdot}, \mu_{k\cdot}) \tag{10}
$$

where $\mu_{k\cdot}$ containing m elements $(\mu_{k1}, \mu_{k2}, ..., \mu_{km})$ and $dist(\cdot, \cdot)$ can be the sum of squared $\sum_{j=1}^{m} (x_{ij} - \mu_{kj})^2$. The process of finding K clusters is the optimization problem to minimize the sum of the distance between the observations and their centroids. Besides, other methods, such as hierarchical clustering, partitional clustering [29] and modularity-based methods [45] are also robust methods for community detection.

2.5. Motivations and innovations

Although studies on the embedding learning of attributed networks and community detection via clustering have been extensively conducted, yielding rich and fruitful results, some considerations are still required when applying these methods to analyze real data. For instance, determining the attributed network embedding dimension remains challenging. Many studies related to attributed network embedding learning have sometimes defined the embedding dimension based on the authors' experience (e.g., 100, 200, or 300 dimensions) [10]. Meanwhile, various metrics for this purpose, such as the cross-validation error [44] and a new metric proposed by [11], have also been developed.

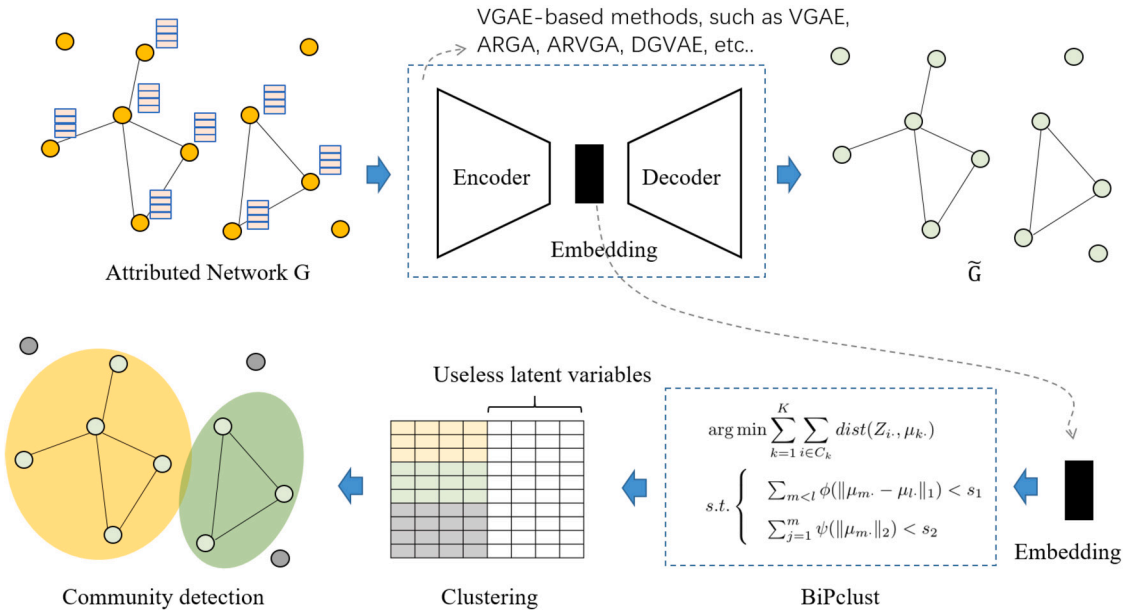

Fig. 2. Framework of detecting communities through VGAE-based methods and bi-direction penalized k-means (BiPClust).

Despite the success of previous studies in dimension selection, further investigation into this problem for different applications is still necessary. The number (or dimension) of the embedding influences, to varying degrees, not only the accuracy of downstream learning tasks, but also the interpretation of the embedding to understand how it works. Similarly, the number of communities is also unknown to some extent, making it challenging for practitioners without a background or knowledge of the application. Consequently, both unknown parameters can hinder the application of community detection in real data analysis. Thus, this study aims to develop a learning framework that automatically selects the embedding dimension and determines the number of communities based on VGAE-based method, k-means and regularized technique. To the best of our knowledge, this study is the first to combine these techniques, offering the following advantages:

- (1) it addresses the challenges of dimension selection and the determination of the number of communities through the automatic optimization of regularized k-means;
- (2) it provides both the computational algorithm and the statistical theorems as evidence, demonstrating that BiPClust can counteract the effects of redundant embedding and determine the unknown number of communities;
- (3) it established the existence of asymptotic estimation in our proposed k-means clustering method with fused lasso penalty, along with its selection consistency. The proposed clustering process is also applicable in the case of diverging dimensions;
- (4) it applies the algorithm for community detection to four benchmark datasets and syndicated investment networks in China to substantiate our theoretical analysis and show that BiPClust outperforms other methods.

3. Community detection based on VGAE-based methods and bi-direction penalized k-means

3.1. Overview of the learning framework

Our goal is to detect communities based on attributed networks. To achieve this goal, the framework comprises two components (shown in Fig. 2): VGAE-based methods [20] used to encode the attributed network into vectorized data, and the **bi-direction penalized k-means clustering algorithm (BiPClust)** used to detect communities based on the embedding learned from the first component.

Previous studies have demonstrated the effectiveness of VGAE-based methods in encoding attributed networks into new representations, thereby enhancing the performance of downstream tasks. Therefore, we utilize the VGAE-based method to transform the unstructured attributed network into structured data (embedding) within our framework. It processes the unstructured data, enabling the clustering algorithm to handle the processed data. In embedding learning, dimension selection is a challenging yet crucial task that significantly influences how much information is retained from the original data in the processed data. Selecting the right dimension is essential, as a large dimension increases computational costs and the complexity for downstream tasks, while a small dimension may result in the loss of useful information.

Although dimension selection can be implemented via cross-validation for hyperparameter tuning, this process can be time-consuming. Moreover, despite advanced dimension selection measures, they may not perform optimally in the embedding learning of the attribute graph. To mitigate computational costs in the embedding learning process, a good strategic approach is to skip the

dimension selection process and leave it to the downstream task. Additionally, opting for a relatively high-dimensional embedding is recommended to retain rich information from the attribute graph, including network topology and node attributes. However, this choice introduces low-informative representations in the downstream learning task, affecting task performance. Therefore, clustering algorithms need to counteract the effects of low-informative features to achieve a robust result.

In addition to the issue caused by informative embedding, the number of communities must be set before using conventional clustering algorithms for community detection. However, in real applications, the exact number of communities or clusters is often unknown and depends on the specific data. In some cases, it cannot be easily determined prior to data analysis. Similarly, we anticipate the algorithm's capability to automatically detect communities with an inexact, relatively large number of clusters. In essence, when provided with an imprecise number of clusters, the algorithm should possess the ability to identify the optimal number of clusters. This allows for the grouping of observations with similar patterns within a cluster and distinct patterns between different clusters, enhancing the algorithm's adaptability to real-world scenarios.

3.2. Bi-direction penalized k-means clustering algorithm (BiPClust)

To solve these two issues, we propose a bi-direction penalized k-means clustering algorithm (BiPClust). Given an embedding in matrix form $Z = (Z_{1\cdot}, Z_{2\cdot}, ..., Z_{n\cdot})^T$ that has n observations, ($Z_{i\cdot} = (z_{i1}, z_{i2}, ..., z_{ip})^T$ is the ith observation). Alternatively, Z can be also represented as the matrix of features $(Z_{\cdot 1}, Z_{\cdot 2}, ..., Z_{\cdot p})$, where $Z_{\cdot j} = (z_{1j}, z_{2j}, ..., z_{nj})^T$ is the jth feature. The idea of a regularization technique is to construct a fused lasso penalty to restrain the difference between any two centroids to control the number of clusters [25,42]. If two centroids are close enough, they will be regarded as the same centroid, and then the two clusters they belong to will be merged into a single cluster. Besides constraining the number of clusters via the bound of the fused lasso, we also restrain centroids of features by a group lasso penalty [48,33], which constrains the value of the same dimension vector in multiple clusters to control useless features and achieve a sparse model. Without loss of generality, let $\phi(\|\mu_{m\cdot} - \mu_{1\cdot}\|_1)$ $(m, l = 1, 2, \ldots, K; m \neq l)$ be the fused lasso penalty [34] and $\psi(\|\mu_{m\cdot}\|_2)$ be the group lasso penalty. The BiPClust is defined as

$$\begin{aligned} &\arg\min \sum_{k=1}^{K} \sum_{i \in C_k} dist(Z_{i\cdot}, \mu_{k\cdot}) \\ &s.t. \begin{cases} \sum_{m<l} \phi(\|\mu_{m\cdot} - \mu_{l\cdot}\|_1) < s_1 \\ \sum_{j=1}^{m} \psi(\|\mu_{m\cdot}\|_2) < s_2 \end{cases} \end{aligned} \tag{11}$$

where s_1 and s_2 are the tuning parameter. Compared with the k-means method in (2), the new model in (11) adds the fused lasso penalty of constraint, the distance between cluster centers, and group lasso penalty of constraint characteristics. Under the constraint of penalties, clusters will be merged if their centroids are close enough and some low informative features that interfere with clustering will be controlled.

In this paper, we specify the penalty function as the combination of fused lasso penalty and group lasso penalty, which are given by $\phi(\|\mu_{m\cdot} - \mu_{l\cdot}\|_1) = \|\mu_{m\cdot} - \mu_{1\cdot}\|_1 = \sum_{j=1}^{d_0} |\mu_{mj} - \mu_{lj}|$ and $\psi(\|\mu_{m\cdot}\|_2) = \sqrt{\sum_{k=1}^{K_0} \mu_{kj}^2}$), respectively. Note that the input embedding Z is standardized with zero mean ($\frac{1}{n}\sum_{i=1}^{n} z_{ij} = 0$) along each feature before clustering. According to the KKT (Karush–Kuhn–Tucker) condition [2], an optimal centroid set $\{\mu_1^\star, \mu_2^\star, ..., \mu_{K_0}^\star\}$ can be obtained through optimizing the regularized k-means clustering algorithm,

$$\arg\min \sum_{k=1}^{K} \sum_{i \in C_k} dist(Z_{i\cdot}, \mu_{k\cdot}) + \lambda_1 \sum_{m<l} \phi(\|\mu_{m\cdot} - \mu_{1\cdot}\|_1) + \lambda_2 \sum_{j=1}^{m} \psi(\|\mu_{m\cdot}\|_2) \tag{12}$$

where λ_1 and λ_2 are the tuning parameters. Evidently, with two penalties, the algorithm is able to handle embeddings with redundant dimensions and also detect the number of clusters with an inexact number of clusters K.

3.3. Theorem analysis

Let the entry L_{ik} of the assignment matrix L have a value 1 if $Z_{i\cdot}$ belongs to the cluster C_k. For $1 < k \leq K$, let D^k be a $(k-1)\times K$ matrix, and

$$D^k_{(k-1)\times K} = \left[-\mathbf{I}_{(k-1)}, \mathbf{1}_{(k-1)}, \mathbf{0}_{(k-1)\times(K-k)}\right], \tag{13}$$

where $\mathbf{I}$ is the identity matrix, $\mathbf{1}$ is a column vector with all elements 1, and $\mathbf{0}_{(k-1)\times(K-k)}$ is $(k-1)\times(K-k)$ zero matrix. For a fixed L, we transform the regularized k-means clustering into the following form

$$\arg\min \sum_{j=1}^{p} \left\{ \frac{1}{n}(Z_{\cdot j} - L\mu_{\cdot j})^T (Z_{\cdot j} - L\mu_{\cdot j}) + \sum_{k=2}^{K} \psi\left(D^k \mu_{\cdot j}\right) + \varphi(\mu_{\cdot j}) \right\} \tag{14}$$

where

$$\psi(D^k \mu_{\cdot j}) = \lambda_1 ||D^k \mu_{\cdot j}||_1, \quad \text{and} \quad \varphi(\mu_{\cdot j}) = \lambda_2 ||\mu_{\cdot j}||_2.$$

Thus, according to the definition, for $1 < k \leq K$, we have

$$D^k \mu_{\cdot j} = \left(\mu_{kj} - \mu_{1j}, ..., \mu_{kj} - \mu_{(k-1)j}\right)^T. \tag{15}$$

Let $\tilde{\mu}$ be the estimated cluster centers from the standard K-means clustering. Then, the optimization problem (14) can be simplified to

$$\arg\min \sum_{j=1}^{p} \left\{ \frac{1}{n}(Z_{\cdot j} - L\mu_{\cdot j})^T (Z_{\cdot j} - L\mu_{\cdot j}) + \sum_{k=2}^{K} \dot{\psi}(D^k \mu_{\cdot j}) + \dot{\varphi}(\mu_{\cdot j}) \right\}, \tag{16}$$

where $\dot{\psi}(D^k \mu_{\cdot j}) = \lambda_1 \frac{||D^k \mu_{\cdot j}||_1}{||\tilde{\mu}_{\cdot j}||_2}$, and $\dot{\varphi}(\mu_{\cdot j}) = \lambda_2 \frac{||\mu_{\cdot j}||_2}{||\tilde{\mu}_{\cdot j}||_2}$.

Let Z be a random vector in $\mathbf{R}^p$ with unknown distribution P, and $Z_{1\cdot}, ..., Z_{n\cdot}$ are the sequence of independent and identically distributed (i.i.d.) observations from the population Z. Denote the empirical probability measure as P_n. Then, the regularized K-means clustering is to minimize

$$W(U, P_n) = \int \min_{\mu_{k\cdot} \in U} ||x - \mu_{k\cdot}||_2^2 P_n(dx) + \sum_{j=1}^{p} \left\{ \sum_{k=2}^{K} \dot{\psi}\left(D^k \mu_{\cdot j}\right) + \dot{\varphi}(\mu_{\cdot j}) \right\} \tag{17}$$

over $U = (\mu_{1\cdot}, ..., \mu_{K\cdot})^T$. We solve the optimization problem (17), and denote the estimator as $\hat{\mu} = (\hat{\mu}_{1\cdot}, ..., \hat{\mu}_{K\cdot})^T$. Let $\bar{\mu} = (\bar{\mu}_{1\cdot}, ..., \bar{\mu}_{K\cdot})^T$ be the true cluster centers which minimizes

$$W(U, P) = \int \min_{\mu_{k\cdot} \in U} ||x - \mu_{k\cdot}||_2^2 P(dx). \tag{18}$$

The corresponding assignment matrices of $Z_{i\cdot}, ..., Z_{i\cdot}$ based on $\hat{\mu}$ and $\bar{\mu}$ are $\hat{L}$ and $\bar{L}$, respectively. Suppose that

$$Z_{\cdot j} = \bar{L}\bar{\mu}_{\cdot j} + \epsilon_{\cdot j} \quad \text{for} \quad j = 1, ..., p, \tag{19}$$

where $\epsilon_{\cdot j} = (\epsilon_{1j}, ..., \epsilon_{nj})^T$ has independent components structure with $E\epsilon_{ij} = 0$, and finite second moment.

In addition, we write $f(n) = O(g(n))$ if $f(n) \leq cg(n)$ for some constant $0 < c < \infty$. The notation $f(n) \asymp g(n)$ means that $f(n) = O(g(n))$ and $g(n) = O(f(n))$.

We obtain the asymptotic estimation of regularized K-means clustering, which is based on the combination of fused lasso and group lasso penalties.

Theorem 1. *Let $Z_{i\cdot}$ be the vector of optimal K-means cluster centers for independent sampling from a distribution P, and $Z_{\cdot j}$ satisfies the model (19).*

- *(i) $\bar{\mu}$ is unique up to relabeling of its coordinates;*
- *(ii) $\int ||Z||_2^2 P(dx) < \infty$;*
- *(iii) The probability measure P has a continuous density f on $\mathbf{R}^p$;*
- *(iv) There exists a dominating function $g(\cdot)$ such that $f(x) \leq g(||x||_2)$, and $r^p g(r)$ is integrable on interval $[0, \infty)$;*
- *(v) Denote $\phi(x, U) = \min ||x - \mu_{k\cdot}||_2^2$. Let Γ be the second derivative matrix of the mapping $U \to P\phi(\cdot, U)$ for distinct centers $\mu_{l\cdot}$. Matrix Γ is positive definite matrix at $U = \bar{\mu}$.*

If $\lambda_1 \asymp \lambda_2$, $n^{1/2}\lambda_l p \to 0$, and $n^{-2}\lambda_l^{-2} p \to 0$ as $n \to \infty$ for $l = 1, 2$, then $\hat{\mu} \to \bar{\mu}$ almost surely and $||\hat{\mu} - \bar{\mu}||_2 = O_p(n^{1/2}\lambda_l p^{-1})$.

We apply the main result in Pollard [28] to the regularized K-means clustering problem. The matrix Γ should be a $Kp \times Kp$ matrix, since the estimated centroid $\hat{\mu}$ is a $p \times K$ matrix in our setting. See Pollard [28] for more details. The proof is shown in **appendix A**.

Let $p_0 < p$. For $j \leq p_0$, we suppose $||\bar{\mu}_{\cdot j}||_2 \neq 0$, otherwise, $||\bar{\mu}_{\cdot j}||_2 = 0$. Let $\mathcal{A} = \{1, ..., p_0\}$ and $\mathcal{A}^c = \{p_0 + 1, ..., p\}$.

To propose the asymptotic selection consistency of the regularized K-means clustering, one more condition is needed.

- (vi) $\arg\min ||Z_{i\cdot} - \mu_{k\cdot}||_2^2$ is unique with probability one.

Theorem 2. *$Z_{i\cdot}$ generated from P, and $Z_{\cdot j}$ satisfies the model (19). Under the assumptions in Theorem 1 and Assumption (vi),*

$$P(||\hat{\mu}_{\cdot j}||_2 = 0) \to 1 \tag{20}$$

for any $j \in \mathcal{A}^c$.

The asymptotic estimation and asymptotic selection consistency of regularized K-means clustering were also studied by Sun et al. [33] under the conditions (i)-(vi). Instead of the method of Sun et al. [33], the regularized K-means clustering considered in the current paper is based on both fused lasso and group lasso penalties, and is more effective for the data that contains some low

informative features. We revisit their results and provide several extensions to fused-and-group lasso penalty. The proof is shown in **appendix B**.

3.4. Computational algorithm

In this section, the augmented lagrangians and the method of multipliers are introduced to solve the optimization problem in (12). Let Θ^k be the difference between μ_k and μ_l ($l \neq k$ and $l = 1, 2, ..., K$), such that

$$\Theta^k = \begin{pmatrix} \theta^k_{11} & \theta^k_{12} & \cdots & \theta^k_{1p} \\ \theta^k_{21} & \theta^k_{22} & \cdots & \theta^k_{2p} \\ \cdots & \cdots & \cdots & \cdots \\ \theta^k_{(k-1)1} & \theta^k_{(k-1)2} & \cdots & \theta^k_{(k-1)p} \\ \theta^k_{(k+1)1} & \theta^k_{(k+1)2} & \cdots & \theta^k_{(k+1)p} \\ \cdots & \cdots & \cdots & \cdots \\ \theta^k_{K1} & \theta^k_{K2} & \cdots & \theta^k_{Kp} \end{pmatrix} \tag{21}$$

To simplify our study, the optimization problem in (12) can be represented in matrix form using a sample matrix Z learned by the VGAE-based method, a clustering centroid matrix μ, and some assign matrices. For example, one assign matrix is L, which allocates samples to a different cluster, such that $L_{ii} = 1$, $L_{ij} = 1$ if ith sample and jth sample belong to the same cluster ($i, j = 1, 2, ..., n$) or set to 0s otherwise. Other assign matrices are $\{D^1, D^2, ..., D^K\}$, used to define the fused lasso penalty. Their definition is similar to the previous one. For ith row, its kth entry is set to 1, i entry is set to -1, and left elements are set to 0s. Therefore, the problem can be rewritten as

$$\begin{aligned} \arg\min\{ & \frac{1}{2}\sum_{j=1}^{m}(Z_{\cdot j} - L\mu_{\cdot j})^T(Z_{\cdot j} - L\mu_{\cdot j}) \\ & + \frac{\lambda_1}{2}\sum_{k=1}^{K}(\theta^k_{\cdot j} - D^k\mu_{\cdot j})^T(\theta^k_{\cdot j} - D^k\mu_{\cdot j}) \\ & + \lambda_3 w_j \|\mu_{\cdot j}\|_2 \\ & + \frac{\lambda_2}{2}\sum_{k=1}^{K}\|\theta^k_{\cdot j}\|_1\} \end{aligned} \tag{22}$$

where λ_1, λ_2 and λ_3 are tuning parameters. In order to deduce the solution, we first need to compute the derivative of (22). Let $f(\mu;\theta)$ be the loss function in (22), gradient of $f(\mu;\theta)$ in the $\mu_{\cdot j}$, and $\theta^k_{\cdot j}$ directions are

$$\begin{aligned} \frac{\partial f(\mu;\theta)}{\partial \mu_{\cdot j}} = & -L^T(Z_{\cdot j} - L\mu_{\cdot j}) \\ & - \lambda_1\sum_{k=1}^{K}(D^k)^T(\theta^k_{\cdot j} - D^k\mu_{\cdot j}) \\ & + \lambda_2\frac{w_j\mu_{\cdot j}}{\|\mu_{\cdot j}\|_2} \end{aligned} \tag{23}$$

and

$$\frac{\partial f(\mu;\theta)}{\partial \theta^k_{\cdot j}} = \lambda_1(\theta^k_{\cdot j} - D^k\mu_{\cdot j}) + \lambda_3 sign(\theta_{\cdot j}) \tag{24}$$

After algebraic calculation, we have

$$(L^TL + \lambda_1\sum_{k=1}^{K}(D^k)^TD^k + \frac{\lambda_2 w_j}{\|\mu_{\cdot j}\|_2}I)\mu_{\cdot j} = L^TZ_{\cdot j} + \lambda_1(D^k)^T\theta^k_{\cdot j}, \tag{25}$$

where $diag(L^TL) = n_k$, which means that entries of L^TL are the sample size of the corresponding cluster the sample belongs to, and

$$\sum_{k=1}^{K}(D^k)^TD^k = \begin{pmatrix} K & 0 & \cdots & 0 \\ 0 & K & 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & 0 & K \end{pmatrix} - \mathbf{1}\mathbf{1}^T \tag{26}$$

Therefore, if $\mu_{\cdot j} \neq 0$, the solution of $\mu_{\cdot j}$ can be updated by the following algorithm step by step until convergence.

Proposition 1. *Given the tuning parameters λ_1, λ_2 and λ_3, according to the KKT condition, $\mu_{\cdot j}^{(s+1)}$ and $\theta_{(kl)j}^{(s+1)}$ are updated respectively by*

$$\mu_{\cdot j}^{(s+1)} = \begin{cases} \frac{L^T Z_{\cdot j} + \lambda_1 (D^k)^T \theta_{\cdot j}^{k,s}}{(n_k + \lambda_1 (K-1)) \| L^T Z_{\cdot j} + \lambda_1 (D^k)^T \theta_{\cdot j}^{k,s} \|_2^2}, & \text{if } \| L^T Z_{\cdot j} + \lambda_1 (D^k)^T \theta_{\cdot j}^{k,s} \|_2^2 > \lambda_2 w_j \\ 0, & \text{otherwise} \end{cases} \tag{27}$$

and

$$\theta_{(kl)j}^{(s+1)} = \begin{cases} S_+(\mu_{kj}^{(s+1)} - \mu_{lj}^{(s+1)}, \frac{\lambda_3}{\lambda_1}), & \text{if } |\mu_{kj}^{(s+1)} - \mu_{lj}^{(s+1)}| > \frac{\lambda_3}{\lambda_1} \\ 0, & \text{otherwise} \end{cases} \tag{28}$$

for $k, l = 1, 2, \ldots, K$ and $k \neq l$; $j = 1, 2, \ldots, p$. $S_+(a, b) = sign(a)(|a| - b)_+$ is the soft-thresholding rule, and $(x)_+ = x$ takes the positive part of x, so that $(|a| - b)_+ = |a| - b$ if $|a| > b$; $(|a| - b)_+ = 0$ otherwise.

The reason why the algorithm works is shown in **appendix C**. According to Proposition 1, centroids of feature j in different clusters will be updated if and only if $\| L^T Z_{\cdot j} + \lambda_1 (D^k)^T \theta_{\cdot j}^k \|_2^2 > \lambda_2 w_j$; otherwise, they will be taken as noisy features and removed from clustering. Similarly, the difference $\theta_{(kl)j}$ between any two centroids along the feature j will be updated if and only if $S|\mu_{kj} - \mu_{lj}| > \frac{\lambda_3}{\lambda_1}$; otherwise, it will be set to zero. These two processes enable the proposed algorithm to select features and determine the number of clusters from high-dimensional data. As a result, we present the whole algorithm for detecting communities in the attribute network Algorithm 1 as follows.

Algorithm 1 Process of detecting communities through the proposed framework.

Input: The attribute networks $\mathbf{G}$, uncertain number of dimension p, uncertain number of clusters K, and tuning parameters λ_1, λ_2 and λ_3.

1: Given uncertain number of dimension p, learning embedding $\mathbf{Z}$ of $\mathbf{G}$.
2: Given tuning parameters λ_1, λ_2 and λ_3, and taking embedding $\mathbf{Z}$ as input, call BiPClust.
3: Initializing clustering centroids $\mu^{\mathbf{0}} = \mu_{1\cdot}^0, \mu_{2\cdot}^0, \ldots, \mu_{K\cdot}^0$.
4: Repeat:
5: Updating $\mu^{(s+1)} = \{\mu_{1\cdot}^{(s+1)}, \mu_{2\cdot}^{(s+1)}, \ldots, \mu_{K\cdot}^{(s+1)}\}$ thought function (27).
6: Removing the useless latent variable j if and only if $\mu_{kj}^{(s+1)} = 0$ for $k = 1, 2, \ldots, K$.
7: Updating $\mathbf{\Theta}^{(s+1)} = \{\theta_{kl\cdot}^{(s+1)}\}$ for $k, l = 1, 2, \ldots, K$ thought function (28).
8: Merging two centroids k and l if and only if $\theta_{klj}^{(s+1)} = 0$ for $j = 1, 2, \ldots, p$.
9: Until $\|\mu^{(s+1)} - \mu^{(s)}\| < \epsilon$ and $\|\mathbf{\Theta}^{(s+1)} - \mathbf{\Theta}^{(s)}\| = \epsilon$.

Output: Clustering results and a subset of latent variables.

3.5. Tuning parameters

The optimal tuning parameters λ_1, λ_2 and λ_3 are obtained when they make the Calinski-Harabasz index (CHI) and the Davies-Bouldin index (DBI). Both indices can be used to evaluate the clustering algorithm when ground truth labels are unknown and the clusters obtained are evaluated using the quantities and the features inherent to the dataset. They are most commonly used to evaluate the quality of the split using a k-Means clustering algorithm for a given number of clusters. A high CHI means the clusters are dense and well-separated, while a low value of DBI indicates the accuracy of clustering results. Therefore, both the CHI and DBI are employed for selecting optimal unknown parameters, such as the tuning parameters λ_1, λ_2 and λ_3 in our study and the unknown number of clusters in k-means or its extensive algorithms. We solve the maximization problem of the CHI and DBI and search for optimal tuning parameters using grid search: first fetching certain tuning parameters through the search space to obtain clustering results, followed by computing the CHI and DBI to validate clustering, and finally obtaining the optimal tuning parameters, which maximize indexes.

4. Experiments and results

In this section, we describe the experiments performed on four benchmark datasets to validate the community detection performance of the proposed approach.

4.1. Benchmark datasets

To evaluate the performance of BiPClust, four benchmark datasets were used in our experiments. The first one is Cora dataset, comprising 2,708 scientific publications classified into seven classes (each class contains many publications coming from the same research field) and the citation network consists of 5,278 edges. Besides the network structure, each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 1,433 unique words seen as the features of publications (more details can be found on the website: https://relational.fit.cvut.cz/dataset/CORA). The second one is Citeseer dataset, which consists of 3,312 scientific publications classified into six classes (or communities). The citation network consists of 4,732 edges and the dictionary consists of 3,703 unique words (more

Table 1
Descriptive statistical analysis of two benchmark data.

Dataset	#Nodes	#Edges	# Average Degree	#Attributes	# Labels	HP-score
Cora	2708	5278	3.89	1433	7	0.84
Citeseer	3312	4660	2.81	3703	6	0.85
Wiki	2405	17981	14.95	4973	17	0.88
Pubmed	19717	44338	4.50	500	3	0.92

details at https://linqs.soe.ucsc.edu/data). Beyond the two datasets, other two networks such as Wiki network (more details can be found in [41]) and Pubmed network (more details at https://pubmed.ncbi.nlm.nih.gov/) are also involved in our experiments to show the performance of proposed method on these special networks. One is a small network but has many communities but another is a large network but has few communities. These four datasets can be seen as attribute networks and the attributes of each node are the bag-of-words extracted from the corresponding paper. These datasets have been used to validate approaches of embedding learning and machine learning for analyzing the attribute network [41,12]. We choose them as benchmark datasets for two reasons. The first one is the number of nodes and communities in our data are similar to Cora and CiteSeer. The second reason is we want to show our method works well on different scales of networks. Pubmed contains 19,717 nodes but only 3 communities which means it has the low signal-to-noise ratio in comparison with Wiki which contains 2405 nodes but 17 communities. The documents in Cora, Citeseer and Pubmed are short texts, while documents in wiki are long texts so that nodes in wiki have more features.

The average degree of nodes in four networks is measured as a simple description of network characteristics. Before analyzing, the Hopkins statistic, which measures the cluster tendency of a dataset, is calculated for each dataset to judge whether it is suitable for clustering. The descriptive statistical analysis of these two datasets is summarized in Table 1.

4.2. Clustering results for the benchmark datasets

Experiment results are conducted based on the setting described as follows. Firstly, the VGAE-based methods are applied to learn the representation of the attribute network given a dimension of embedding (e.g., 16 or 64). By setting different dimensions, we demonstrate the robustness of clustering algorithms in the following steps. We visualize the samples by using t-Distributed Stochastic Neighbor Embedding (t-SNE) [24] (a technique for dimensionality reduction that is particularly well-suited for the visualization of high-dimensional datasets) to show whether the embedding represents the actors well when we know the true label. **Appendix D** shows that the boundaries between the clusters are not completely clear based on VGAE, other VGAE-based methods have similar results. It probably contains noise information for distinguishing samples that needs to be removed during clustering.

Since, unlike BiPClust, VGAE-based methods are unable to directly detect the number of clusters, CH and DBI indexes are also used for that purpose. Following the conclusion in Luxburg [35], we set the clustering number from 2 to $\sqrt{n}$ for the methods that are unable to determine the number of clusters, but empirically set $\sqrt{n}$, where n is the sample size, for the methods that can determine it. To illustrate this, the proposed method may be applied to these datasets with varying numbers of clusters and the resulting clusters compared to the ground truth. Finally, we assess the quality of clustering results via clustering internal validation indices, such as purity, accuracy, F1-score and adjusted rand index (as mentioned in previous studies [42]). For all indices, the larger their value, the better the performance of clustering algorithm. In addition, the number of clusters is also reported to evaluate the ability to determine the unknown number of clusters. According to experimental settings, we evaluate the learned representation by analyzing the clustering results obtained by clustering methods.

According to experimental settings, we evaluate the learned representation by analyzing the clustering results obtained by BiPClust and the other clustering algorithms, namely the classic k-means algorithm (Kmeans), the classic Hierarchical clustering algorithm (HClust), the sparse k-means algorithm (Sparcl) [37], and the penalized clustering algorithm (PClust) [42], and the group-lasso penalized clustering algorithm (GClust) [33]. We repeatedly run each clustering methods 50 times and then compute the average value of all evaluation metrics. The experiment is evaluating the performance of BiPClust in comparison with other clustering algorithms with the obtained embedding matrix. The number of clusters is obtained through optimizing CH and DBI indexes. We only present clustering results of the embedding learned by VGAE and LMAVGAE for two reasons. One is many VGAE-based methods are the extensive methods of VGAE; another is because LMAVGAE performs well on four benchmark datasets (shown in Table A.1 in Appendix) consistent with previous research findings. Thus, the results on four benchmark datasets including both VGAE and LMAVGAE are shown in Tables 2 and 3.

A first analysis of the results for four benchmark datasets reveals that BiPClust performs as well or even better than other clustering approaches given two different dimensions (64 and 16). It shows the highest values for most clustering validation indices, though the purity of HClust, Sparcl and GClust is greater than that of our method in the cases where they obtain more clusters. Secondly, BiPClust is always close to the true number of clusters (e.g., the true clusters of CiteSeer are 6 vs the 7 and 5 clusters found in our results). In addition, BiPClust also performs extremely well in different dimensions of embedding (from 64 to 16), even with less informative representations. Furthermore, the results of BiPClust exhibit a smaller gap for different dimensions compared to other methods, which shows that it is a relatively more robust algorithm. By contrast, PClust and Kmeans are unable to recognize what features are noisy, despite their ability to cluster high-level noisy data. Likewise, the Sparcl approach cannot select features correctly.

Table 2
Evaluation of clustering results based on the learned embedding by VGAE with different dimensions.

Dataset	# of Embedding	Method	# of Clusters	Purity	Accuracy	F1	ARI	NMI
Cora	16	BiPClust	7	65.30%	63.50%	48.13%	37.35%	43.65%
		Kmeans	7.8	65.10%	57.13%	44.13%	33.90%	42.89%
		HClust	7.5	61.86%	57.98%	45.39%	33.71%	40.38%
		Sparcl	8	62.25%	55.51%	41.22%	30.46%	39.85%
		Gclust	9	66.17%	55.23%	42.36%	32.86%	42.51%
		PClust	8.75	64.93%	56.28%	43.07%	33.06%	43.03%
	64	BiPClust	9.6	66.26%	56.80%	44.39%	31.22%	43.64%
		Kmeans	5.6	52.84%	51.11%	38.93%	21.81%	36.52%
		HClust	12.2	61.52%	44.10%	33.42%	18.10%	37.43%
		Sparcl	5.75	43.57%	41.62%	29.82%	13.30%	20.58%
		Gclust	6	60.25%	54.94%	44.46%	28.69%	40.77%
		PClust	7.5	60.08%	54.34%	43.11%	24.30%	42.68%
Citeseer	16	BiPClust	7	52.21%	44.57%	31.61%	16.23%	22.61%
		Kmeans	8.33	49.37%	35.13%	27.13%	13.41%	20.36%
		HClust	8.33	47.67%	35.28%	26.76%	9.94%	19.98%
		Sparcl	8.33	48.42%	35.23%	26.36%	12.52%	19.06%
		Gclust	4	42.65%	41.80%	32.01%	11.16%	18.82%
		PClust	5	44.83%	43.23%	32.06%	10.67%	21.96%
	64	BiPClust	5	45.16%	42.53%	31.35%	10.87%	21.00%
		Kmeans	6	43.33%	39.24%	28.22%	9.11%	18.86%
		HClust	5	37.54%	34.79%	29.59%	4.80%	16.25%
		Sparcl	6	35.38%	32.06%	22.80%	5.17%	9.37%
		Gclust	4.5	40.59%	38.71%	30.09%	7.60%	18.92%
		PClust	7	41.57%	35.75%	29.43%	5.55%	19.74%
Wiki	16	BiPClust	10	46.61%	45.47%	32.83%	24.92%	40.05%
		Kmeans	9	43.59%	42.09%	31.39%	22.90%	38.17%
		HClust	7.67	39.49%	38.37%	29.05%	19.37%	34.56%
		Sparcl	9	43.59%	42.04%	30.97%	22.39%	37.80%
		Gclust	10.67	46.47%	45.09%	32.08%	24.45%	40.04%
		PClust	11.67	47.55%	43.63%	31.11%	23.76%	39.24%
	64	BiPClust	13.75	56.50%	49.62%	35.06%	28.72%	43.95%
		Kmeans	14	54.26%	47.65%	32.12%	25.30%	43.08%
		HClust	9.5	46.39%	43.18%	27.83%	17.47%	38.91%
		Sparcl	14	54.47%	48.13%	32.33%	25.64%	42.86%
		Gclust	13.75	52.69%	47.68%	32.44%	25.34%	42.31%
		PClust	13.25	54.46%	48.32%	33.41%	26.54%	43.29%
Pubmed	16	BiPClust	3	66.65%	66.65%	51.92%	26.01%	26.05%
		Kmeans	3	66.34%	66.34%	51.61%	25.56%	25.74%
		HClust	4	69.93%	59.50%	49.08%	27.60%	28.69%
		Sparcl	3	65.80%	65.80%	50.72%	24.44%	24.37%
		Sparcl	3	66.16%	66.16%	51.40%	25.26%	25.50%
		PClust	3	66.54%	66.54%	51.83%	25.86%	25.95%
	64	BiPClust	3	65.80%	65.80%	52.02%	25.58%	28.24%
		Kmeans	4	69.47%	52.61%	47.15%	24.85%	29.20%
		HClust	5	68.02%	50.51%	42.26%	20.12%	23.80%
		Sparcl	4	61.87%	51.34%	43.19%	17.55%	22.49%
		Sparcl	4	69.47%	52.69%	47.03%	24.71%	29.06%
		PClust	4	67.00%	52.14%	44.70%	21.11%	26.06%

4.3. Clustering analysis of data with different numbers of embedded features

By comparing the clustering results for different numbers of embeddings except 16 and 64, we demonstrate the impact that dimensionality has on the quality of clustering results and the importance of selecting an appropriate number of embeddings for a given dataset. VGAE-based methods are applied to learn the representation of Cora given a dimension of embedding (e.g., 24, 32 and 48) first, and then various clustering methods have performed on Cora and obtained the clustering results to compare the clustering results for different numbers of embedding. Results in Tables A.4 and A.5 show the number of embedded features or the dimensionality of the embedding space can have a significant impact on the clustering performance. However, the proposed new method outperforms other methods regardless of the number of embedded features set, indicating that it is a robust and effective method.

4.4. Clustering analysis of data with different tuning parameters

To illustrate the process of tuning parameters, we perform the proposed clustering method on the Cora dataset with different tuning parameters and evaluate the clustering results using same metrics mentioned before previous section. Our goals are searching

Table 3
Evaluation of clustering results based on the learned embedding by LMAVGAE with different dimensions.

Dataset	# of Embedding	Method	# of Clusters	Purity	Accuracy	F1	ARI	NMI
Cora	16	BiPClust	6.8	69.43%	66.27%	53.88%	43.22%	50.45%
		Kmeans	7.8	68.21%	58.68%	47.97%	37.47%	46.90%
		HClust	7.6	66.49%	62.94%	49.52%	37.73%	45.44%
		Sparcl	7.8	58.80%	50.64%	38.00%	26.14%	36.72%
		Gclust	9	71.62%	57.84%	47.07%	37.83%	48.84%
		PClust	8.6	67.61%	58.09%	46.01%	35.57%	46.41%
	64	BiPClust	9	65.76%	57.21%	45.21%	31.14%	45.30%
		Kmeans	6.4	56.96%	55.66%	42.70%	24.62%	37.53%
		HClust	16.2	63.83%	40.36%	31.52%	16.81%	38.71%
		Sparcl	6.4	37.68%	31.93%	23.84%	7.57%	12.26%
		Gclust	8	61.81%	52.58%	40.04%	26.41%	39.83%
		PClust	7.4	60.33%	54.31%	41.27%	21.33%	42.09%
Citeseer	16	BiPClust	5.67	48.57%	46.92%	33.23%	13.24%	23.50%
		Kmeans	4.67	44.56%	43.24%	32.59%	12.49%	22.44%
		HClust	3	40.44%	40.44%	33.06%	8.88%	18.70%
		Sparcl	4.67	48.09%	46.54%	33.63%	14.76%	21.74%
		Gclust	5	45.49%	44.33%	31.97%	14.42%	20.40%
		PClust	6	46.46%	42.90%	31.90%	12.89%	22.65%
	64	BiPClust	6.67	46.45%	43.36%	32.00%	10.64%	22.67%
		Kmeans	6.33	45.35%	42.21%	31.07%	10.50%	22.25%
		HClust	3	40.01%	40.01%	32.78%	11.85%	18.32%
		Sparcl	6.33	45.30%	41.88%	30.65%	11.39%	21.45%
		Gclust	4.67	43.80%	43.26%	31.95%	10.76%	20.75%
		PClust	7.67	47.07%	40.33%	30.35%	9.91%	23.53%
Wiki	16	BiPClust	15	57.06%	48.86%	35.38%	29.52%	45.23%
		Kmeans	13.5	52.51%	46.82%	33.25%	26.46%	43.53%
		HClust	9.5	47.42%	45.62%	30.56%	21.26%	39.93%
		Sparcl	13.5	52.22%	45.38%	32.32%	25.50%	42.01%
		Gclust	14	54.27%	47.73%	33.57%	27.09%	43.95%
		PClust	13.75	54.23%	48.45%	35.03%	28.77%	44.05%
	64	BiPClust	15	54.95%	47.05%	31.59%	24.91%	42.49%
		Kmeans	15.25	53.49%	42.53%	27.48%	19.21%	40.82%
		HClust	8.75	46.89%	45.80%	27.56%	15.85%	39.01%
		Sparcl	15.25	52.08%	42.42%	26.80%	18.80%	39.29%
		Gclust	15.25	53.88%	44.94%	29.10%	21.83%	41.33%
		PClust	15.25	54.35%	45.53%	29.31%	21.06%	42.23%
Pubmed	16	BiPClust	3	65.45%	65.45%	52.03%	25.44%	25.90%
		Kmeans	3	65.42%	65.42%	52.02%	25.41%	25.91%
		HClust	3	55.87%	54.60%	45.08%	13.06%	19.55%
		Sparcl	3	65.19%	65.19%	52.07%	25.32%	26.46%
		Gclust	3	65.29%	65.29%	51.84%	25.18%	25.75%
		PClust	3	65.41%	65.41%	52.00%	25.39%	25.89%
	64	BiPClust	4	67.67%	52.28%	45.27%	21.95%	26.93%
		Kmeans	5	62.29%	45.86%	38.74%	15.25%	21.50%
		HClust	6	67.78%	49.45%	41.22%	19.30%	23.71%
		Sparcl	5	64.01%	42.69%	39.33%	16.72%	23.94%
		Gclust	4	67.56%	52.69%	45.14%	21.76%	26.61%
		PClust	4	67.62%	52.37%	45.25%	21.90%	26.89%

the optimal parameters (including λ_1, λ_2 and λ_3) to maximize CH and DBI indexes using the strategy of a grid search. First, we define a grid of hyperparameters to search over. Second, for each combination of hyperparameters, we apply the clustering algorithm to our dataset and compute the CH and DBI indices. Then, we store the results of each combination of hyperparameters, along with the corresponding CH and DBI scores. Next, we analyze the results and identify the hyperparameters that maximizes the CH and DBI indices. Finally, we select the optimal hyperparameters that produce the highest CH and lowest DBI scores and use them to perform clustering on the full dataset. Results in Table A.6 indicate the proposed method can find the suitable combination of hyperparameters to optimize CH and DBI scores.

4.5. Computational complexity analysis

According to Algorithm 1, the computational complexity of BiClust depends on four operations. The most important two steps are updating μ and are updating Θ. Because dimension of Θ is determined by the initial number of clusters which is smaller than $\sqrt{n}$ and Θ is calculated by subtraction operation, the computational complexity of updating Θ can be ignored. In this case, analyzing

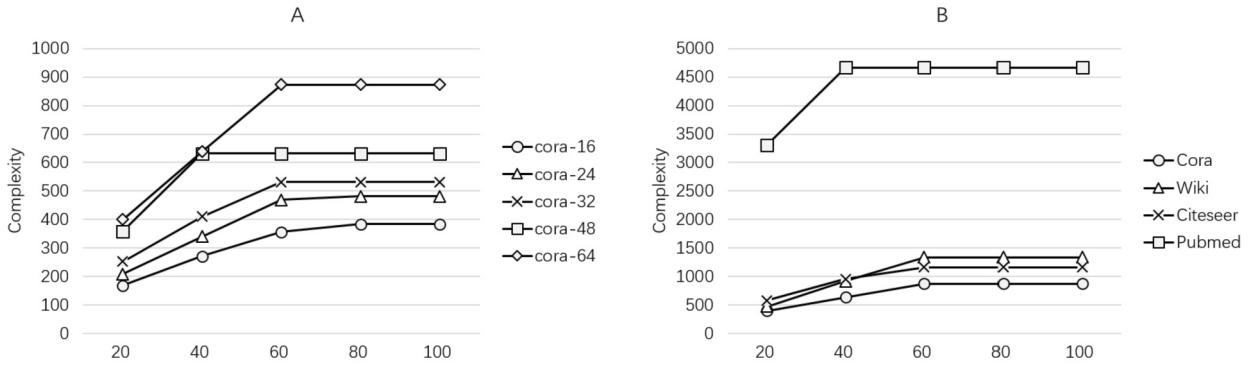

Fig. 3. Computational complexity analysis with different iterations.

Table 4
Brief description of syndicated investing networks.

Dataset	#Nodes	#Edges	# Average Degree	#Attributes	HP-score
2001-2005	130	162	4.98	28	0.72
2006-2010	484	1069	4.42	28	0.79
2011-2015	1047	2768	5.29	28	0.83

the complexity of updating μ can understand the whole computational complexity of the proposed method. Observing the updating equation $\frac{L^T Z_{\cdot j}+\lambda_1 (D^k)^T \theta_{\cdot j}^{k,s}}{(n_k+\lambda_1(K-1))\| L^T Z_{\cdot j}+\lambda_1 (D^k)^T \theta_{\cdot j}^{k,s}\|_2^2}$, it seems a little complexity but the most time consuming is $(D^k)^T \theta_{\cdot j}^{k,s}$; and because L is the assign matrix, $L^T Z_{\cdot j}$ will be not changing in the process of estimating μ, this part does not need to be calculated each iteration. Therefore, without considering the role of two penalties, the most time consuming part has computational complexity of $O(IK^2p)$ where I is the maximum number of iterations, K is the number of clusters and p is the dimensionality of the data. Through observing the working process of the proposed method while analyzing benchmark datasets, we found: (1) the running time increases linearly with the number of iterations and the proposed algorithm converges after reaching a certain number of iterations (shown in Fig. 3); (3) the larger the embedding dimension, the longer it takes (shown in Fig. 3-A); and (2) the larger the dataset size, the longer it takes (shown in Fig. 3-B).

5. Application

In this section, we use the proposed approach to exploit the syndicated investment network and detect communities to trace the changes in the venture capital (VC) market in China from 2001 to 2015. Furthermore, we obtain some interesting results related to the Chinese VC market by analyzing the changing structure.

5.1. Data collection

In the subsequent analysis, we employ the proposed method to analyze the syndicated investment network and identify communities, offering insights to track changes in the VC market in China. In particular, we focus on the syndicated investment networks generated from 2001 to 2015. The data have been collected from major VC databases, including ChinaVenture, Zero2IPO, and the Venture Capital Research Institute's annual report, which contains data regarding all public investments and relevant indexes in the VC field in China. Building on a previous study [13], we employ non-overlapping 5-year windows to construct the syndicated investment network. This assumption is grounded in the belief that such relationships erode over time. The chosen window allows adequate time to identify VCs' preferences for syndication while avoiding excessively long periods that may contain stale information. VCs are indirectly linked through their joint investments in one or more firms. Each edge of the network is coded as 1, signifying a connection between two firms through a joint investment in a specific investment round, while 0 indicates no such connection [43]. Thus, three networks are defined based on three time windows to describe changes in the communities. The first window gathers 130 VCs, containing 162 co-investing activities in three stages (i.e., the initial, expansion, and seed stages) from 2001 to 2005. The second one collects 484 VCs with 1,069 co-investing activities from 2006 to 2010, and the last one includes 1,047 VCs and 2,768 co-investing activities from 2011 to 2015. More details on the three syndicated investment networks are presented in Table 4.

In addition to syndicated networks, VCs' attributes derived from investment events are also considered in our analysis. Each event indicates that a VC firm has invested in a startup. Investment information includes which startup a VC has invested in, when and where, which industry the startup belongs to, and the investing period (such as initial, expansion, and seed stages), and round (A, B, C, D, E, F, and G rounds). Usually, most investors prefer to invest in several firms, industries, areas, stages, and rounds to diversify investment risks. Some VCs have more resources than others. They tend to operate on a larger scale and have greater experience

Table 5
Clustering results of venture capital firms.

Dataset	Dimension	Selected Dimension	Clusters	Number of each clusters
2001-2005	16	14	6	1(28), 2(34), 3(16), 4(10), 5(20), 6(22)
	64	53	6	1(26), 2(33), 3(19), 4(10), 5(18), 6(24)
2006-2010	16	13	7	1(76), 2(88), 3(85), 4(32), 5(42), 6(100), 7(61)
	64	44	6	1(114), 2(93), 3(83), 4(66), 5(45), 6(83)
2011-2015	16	15	5	1(158), 2(330), 3(212), 4(223), 5(124)
	64	36	5	1(177), 2(253), 3(222), 4(241), 5(154)

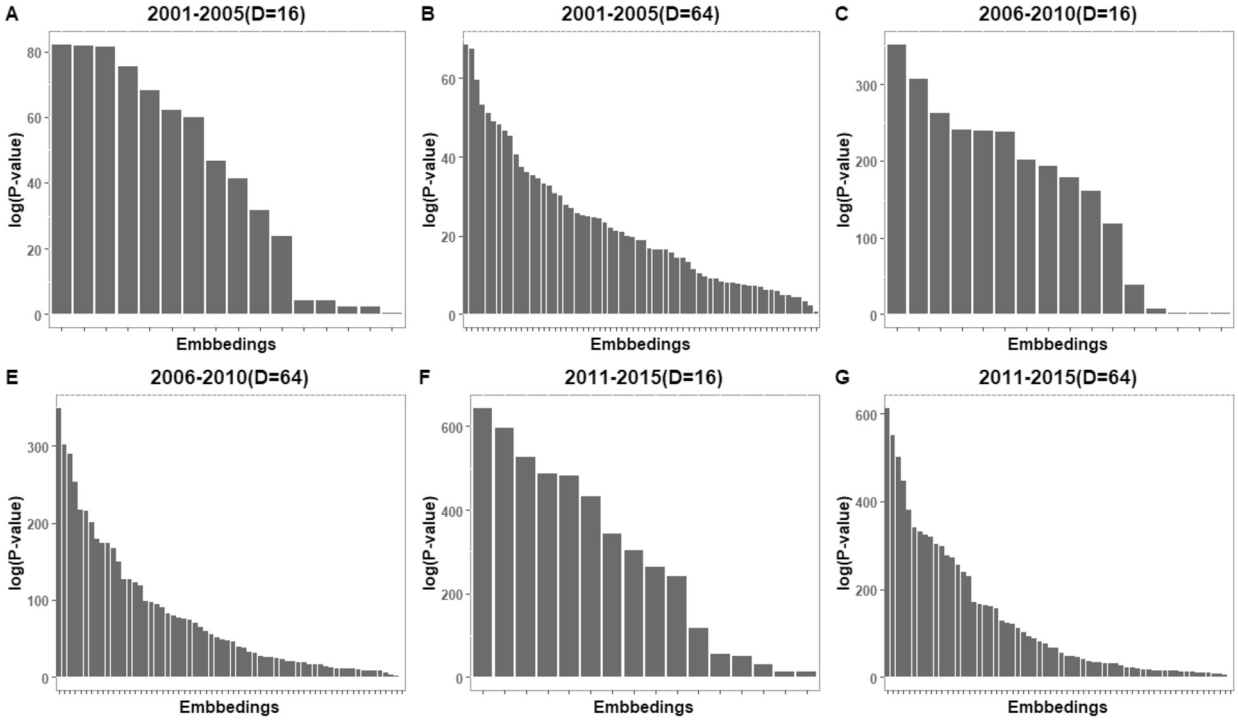

Fig. 4. Importance of embedding in different clustering results.

than those who can invest only small amounts. Therefore, such attributes related to scale and experience are vital to describe VCs [43], including 28 other features. A brief description of them is presented in **appendix E**.

5.2. Clustering results of real data

Our analysis is threefold: first, we identify the communities of three syndicated networks and obtain the clustering results; second, we analyze the changes in the communities by comparing the vertical and horizontal communities; and third, we illustrate our findings by analyzing the communities. Because both the embedding dimension and the number of communities are unknown, BiPClust is introduced to address these issues in community detection for attributed syndicated networks. The optimal tuning parameters λ_1, λ_2 and λ_3 are selected using both CHI and DBI, similar to the previous analysis. The given dimension (Dimension) and the selected dimension of the embedding, the number of clusters, and the sample size in each sub-cluster are listed in Table 5, showing different clustering results for the three time windows. Clustering results of BiPClust are visualized using t-SNE to show whether actors can be classified into the correct community [24]. The visualization result is shown in Fig. 5, indicating compact and well-separated clusters are obtained using BiPClust.

As indicated in Table 5, VCs are categorized into six or seven categories in the first two periods (2001-2005 and 2006-2010), but they are categorized into five groups in the third period (2011-2015) because of the relatively few collaborations in the first two periods compared with that in the third period, which led to sparse networks in the former case and a denser one in the latter case. In addition to its performance in clustering results, BiPClust performs well in determining the number of clusters under different dimensions (e.g., 16 and 64), indicating that BiPClust can select informative embedding and counteract the effects of noisy embedding and that the selected dimension is smaller than the given dimension. We further analyze the difference between embeddings through F-statistics, a value obtained from an ANOVA test used to verify if the means between different clusters are significantly different. The log of the p-value is used to quantify the importance of the embedding, as shown in Fig. 4. The importance

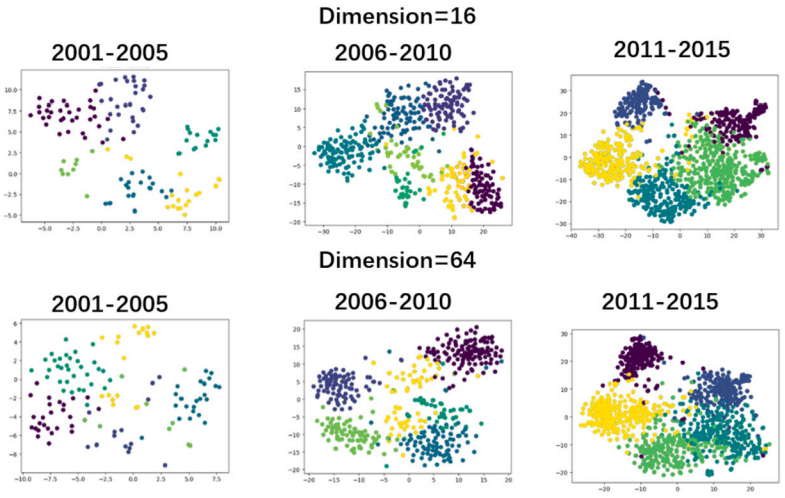

Fig. 5. t-SNE plot visualizing cluster assignments of VCs in real data.

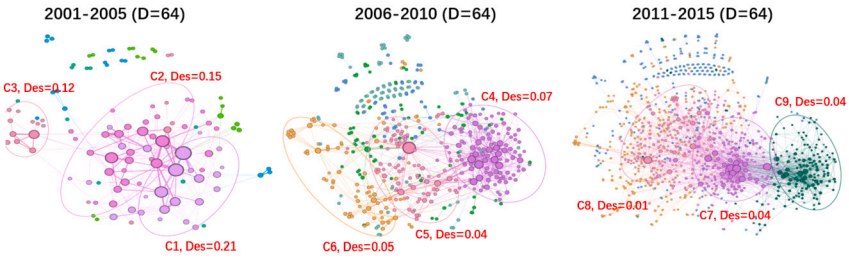

Fig. 6. Changes in community structure of venture capital market in China.

of the embedding increases with the increasing log of the p-value. The figure shows that some embeddings are less informative to represent the difference between clusters. In addition, the new algorithm removes less informative embedding and retains more informative ones when clustering. Finally, the t-SNE visualization graph shown in Fig. 5 provides an insight into the extent of clarity of the boundaries among the clusters generated by BiPClust for the embeddings of size 64 and 16. The clear boundaries seen in Fig. 5 further demonstrate that this algorithm performs well in clustering even without providing an exact number of clusters, unlike the traditional methods that require the cluster number to be determined in advance. It also shows the advantage of determining the number of clusters by penalizing the loss function.

5.3. Changes in communities

In this section, we analyze the results to illustrate changes in the community structure from 2001 to 2015 in China based on the embedding of size 64. Fig. 6 shows that the syndicated investment in each community becomes more intense over time, consistent with the development of the VC market in China. Looking back at its history, China started late, in the early 1990s, began to develop after 2005 and has experienced drastic growth since 2010, becoming the second-largest VC market in the world. In the early years, domestic VC firms were not considerably successful owing to the shortage of diverse investment channels. Nevertheless, VCs began to rapidly grow after the establishment of the Small- and Medium-Sized Enterprise Board in 2004. Thereafter, VCs have developed rapidly, benefiting from the Internet. This corroborates the increasing number of VCs and syndicated investments seen in Table 4. To simplify our study, we identify some dominating communities according to their density and location in the network. We calculate the density of each community, defined as $Den = \frac{2*|E|}{n*(n-1)}$, where $|E|$ represents the number of edges and n represents the number of nodes in a community, also presented in **Appendix F**. We denote the dominant communities in three networks from 2001 to 2015 as $\{C_1,C_2,...,C_9\}$ and the remaining ones as $\{O_1,O_2,...O_8\}$. In particular, C_1, C_2, and C_3 are the three most connected communities in the syndicated investment network for the first period (2001-2005). Similarly, C_4, C_5, and C_6 dominate the network in the second period (2006-2010), while C_7, C_8, and C_9 are the most important ones in the third period (2011-2015). Despite the increasing scale and connections of the communities, their density decrease over time because the increasing speed of edges is slower than that of the nodes. However, the density of dominating communities, shown in Fig. 6, is always greater than others.

To illustrate the difference between two communities for any two periods, we quantify the change tendency in VC communities from 2001 to 2015: we define two metrics as the indices of similarity and dissimilarity between two sets. The first one is $\frac{C_i(t-1)\cap C_j(t)}{C_i(t-1)}$, called the stability or unchangeability, illustrating the ratio of common elements in both sets $C_i(t-1)$ and $C_j(t)$ divided by the

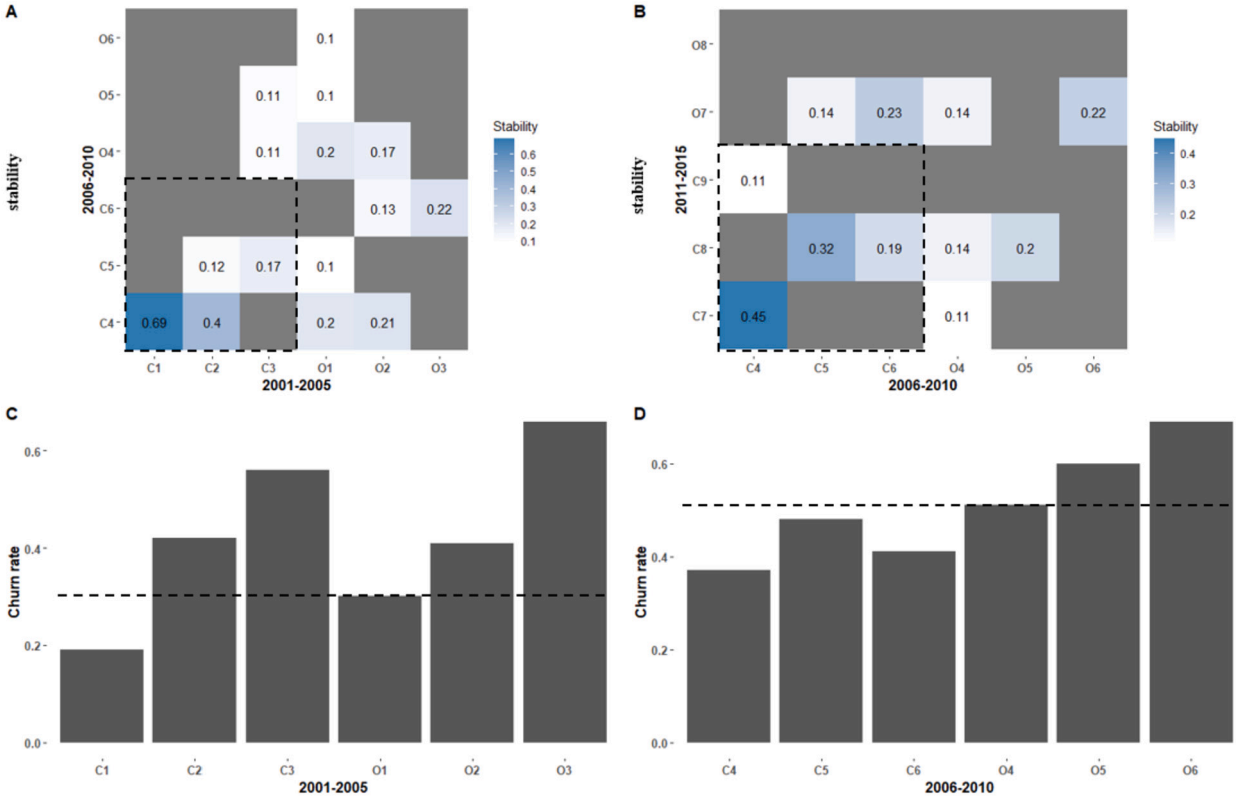

Fig. 7. Changes in community structure of venture capital market in China.

number of $C_i(t-1)$, measuring the extent to which C_i and C_j overlap, and adapting to evaluate similarity and dissimilarity between two communities in different periods. In addition, it quantifies the number of VCs retained in a set from the previous period (t-1) to the current one (t). The second one is $\frac{C_i(t-1)-C_i(t-1)\cap C(t)}{C_i(t-1)}$, where C(t) is the union set of all communities in the period t, called the changeability or the churn rate, and measures how many VCs are absent in the next period (t), but exist in $C_i(t-1)$ in the previous period (t-1), indicating changes in a certain community.

Owing to their drastic growth since 2010, VCs and connections in the whole network become denser over time. We denote the communities with numerous connections as dominant communities (C_1, C_2,...,C_9) unlike communities with sparse connections (O_1,O_2,...,O_9). Fig. 7.A shows that many members in communities (such as C_1, C_2, O_1, and O_2, particularly C_1 and C_2) in the first period (2001-2005) have moved to community C_4 in the second period (2006-2010). C_5 has more members in common with C_2 and C_3 than with the others. The main members of C_6 come from O_3. Moreover, we find that most members of C_4 remained in C_7, a few of them have moved to C_9, and the others disappeared, while many members of C_8 came from C_5 and the remaining members moved to other communities with fewer collaborations, similar to C_6 in Fig. 7.B. Evidently, considerable changes in community structure can be observed from 2001 to 2015. Some previously existing communities disappeared, some merged into a single community, and some new ones were born. Irrespective of the types of changes, dominant communities display greater stability. For example, C_1 and C_4, which share the most mutual members, have a stability rate of 0.69. As for other cases, we have C_2 and C_4 (stability = 0.4), C_4 and C_7 (stability = 0.45), and C_5 and C_8 (stability = 0.32), indicating that dominant communities are more likely to retain common VCs from one period to another and thus are the most stable among the communities. Typically, a more stable community has a lower churn rate. The churn rate for C_1 and C_4 is the lowest. In addition, considering the syndicated investment network for the 2011-2015 period as an example, we calculate the average degree of centrality (degree centrality), degree centralization, clustering coefficient (Clustering), closeness centrality (Closeness), and density of each community. The results present in Table 6 show that the connections of dominant communities (C_7, C_8, and C_9) are more intense than those of the others (O_7 and O_8). All metrics are the greatest.

Furthermore, we obtained the top-50 high-status VCs by calculating the eigenvector centrality and found that all of them involved dominant communities (66% from C_7, 8% from C_8 and 26% from C_9). Moreover, the top-5 high-status VCs in each dominant community are shown in Fig. 8; further, the neighbors' degree centrality (Neighbors' DC) and the efficiency of these VCs are shown in Table 7. The neighbors' degree centrality describes the importance of the partners of a VC; the higher it is, the more partners this VC's partners have, and the more important they are in the network. The efficiency of a VC's ego network based on the concept of redundancy measures the extent to which it is connected to its neighbors. A greater efficiency of a VC indicates less redundancy and fewer connections with its neighbors. Furthermore, we distinguished VC ownership based on foreign and domestic firms in the

Table 6
Descriptive statistic for communities in the syndicated investment network 2011-2015.

Community	Degree Centrality	Degree Centralization	Clustering	Closeness	Density
C7	7.6271	0.3010	0.4802	0.3805	0.0430
C8	3.5336	0.2179	0.3310	0.2406	0.0140
C9	6.9221	0.2323	0.3232	0.3066	0.0452
O7	2.1441	0.0450	0.3139	0.0210	0.0097
O8	1.0788	0.0123	0.1349	0.0049	0.0045

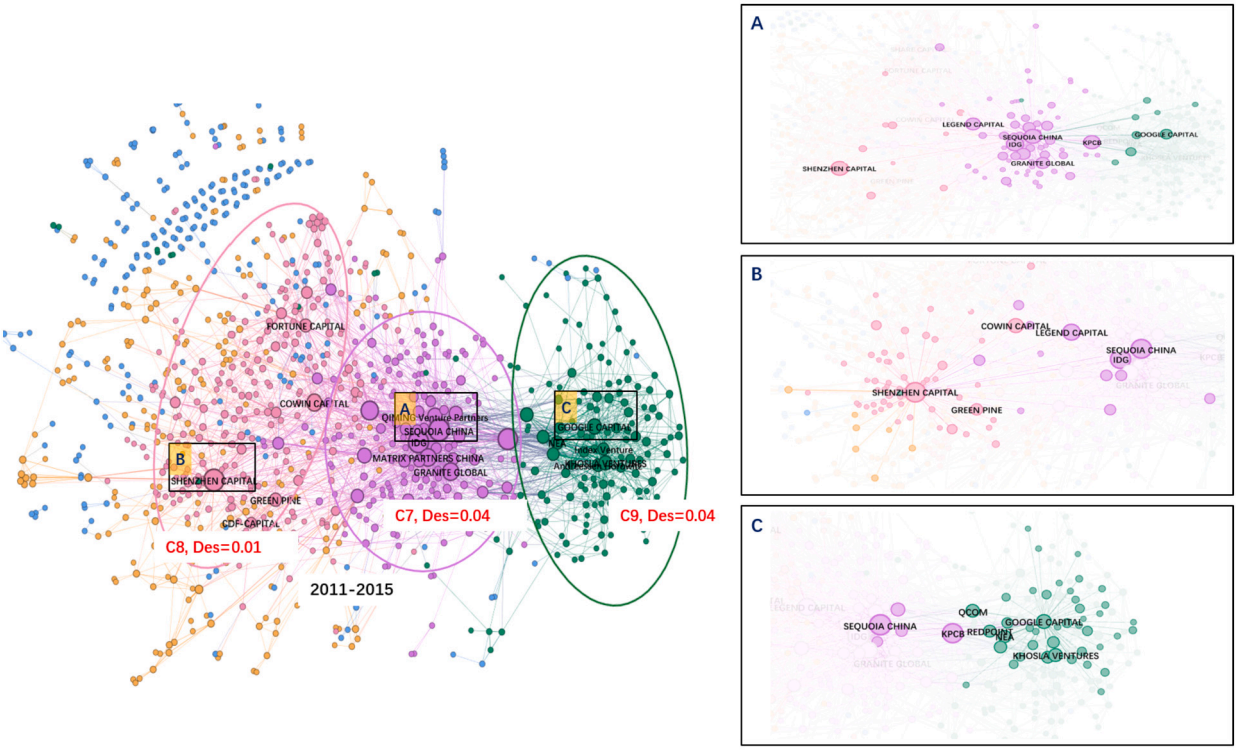

Fig. 8. Leading VCs in different communities in 2011-2015.

context of the Chinese venture market. Typically, C_9 is a foreign community with foreign VCs accounting for 83.77% compared with C_8, which is a domestic community with domestic VCs accounting for 94.5%. Combined with Table 7, this result shows that the average efficiency of high-status VCs in C_8 (0.9261) is larger than C_7 (0.8532) & C_9 (0.8087) and the average neighbors' degree centrality of C_8 is the smallest (0.0136), indicating that the cooperation between high-status VCs in the foreign community is closer than that in the domestic community and that the partners of high-status VCs in the domestic community have a lower degree centrality (less prominent in the network). This can also be seen from Fig. 8: the connections in C_8 are not as intense as in C_9, and they diverge from high-status VCs (such as Shenzhen Capital, COWIN Capital, GREEN PINE, CDF-CAPITAL, and FORTUNE Capital) as its core is surrounded by some small VCs in C_8.

6. Conclusion and discussion

This paper proposes a learning framework for community detection comprising two components: embedding learning and clustering. This study is the first to combine VGAE to encode attributed networks, and k-means and bi-regularized techniques for community detection. Our approach (1) solves dimension selection and determines the community number automatically, (2) offers both a computational algorithm and statistical theorems to substantiate its efficacy in mitigating the impact of redundant embedding and determining the unknown number of communities, and (3) evaluates it based on four benchmark datasets and a practical scenario. Most validation indices are at the best level. Furthermore, we show that the number of embedded features or the dimensionality of the embedding space can considerably affect the clustering performance, but our proposed method can alleviate the influence and outperform other methods if the number of embedded features is unknown.

The proposed new method was applied to detect communities in syndicated investment networks in China from 2001 to 2015. It can employ both network topology and attributes. Our analysis is threefold: first, we identify communities of syndicated networks in three time periods and present the clustering results; second, we explore the evolution of communities through horizontal and

Table 7
Top-5 High-status VCs in each community of the syndicated investment network 2011-2015.

Community	VC	Neighbors' DC	Average	Efficiency	Average
C_7	IDG	0.0205	0.0207	0.8764	0.8532
	SEQUOIA CHINA	0.0186		0.8907	
	MATRIX PARTNERS CHINA	0.0206		0.8496	
	GRANITE GLOBAL	0.0227		0.8252	
	QIMING Venture Partners	0.0212		0.8242	
C_8	SHENZHEN CAPITAL	0.0110	0.0136	0.9667	0.9261
	COWIN CAPITAL	0.0162		0.9273	
	GREEN PINE	0.0149		0.9224	
	CDF-CAPITAL	0.0110		0.8930	
	FORTUNE CAPITAL	0.0148		0.9211	
C_9	GOOGLE CAPITAL	0.0192	0.01887	0.8126	0.8087
	Andreessen Horowitz	0.0188		0.7731	
	Index Venture	0.0200		0.7722	
	KHOSLA VENTURES	0.0168		0.8514	
	NEA	0.0196		0.8186	

vertical comparisons; and third, we elaborate on the findings by analyzing the communities in detail. First, among the notable findings from our analysis, we observed that the clustering results align consistently with the evolution of the VC market in China. The t-SNE plots clearly depict well-separated communities detected by BiPClust. Second, China's VC network structure changed considerably between 2001 and 2015 with the rapid expansion of the VC market. The connections of VCs became salient over time. From one period to another, existing communities might become extinct or strengthened through cooperation and even merged into a single community, while some new communities might be born. Dominating communities displayed higher stability than other communities. In addition, they were more likely to retain common VCs over two consecutive periods and had relatively low churn rates. Finally, dominating communities included more high-status VCs. Compared with the domestic community, the community dominated by foreign VCs adopted more joint investment strategies.

However, further work remains to be done in the future. First, more experiments using embedding methods other than the VGAE must be performed to prove the universality of the model and improve the hyperparameter tuning method to make the model more robust and effective. Second, the relationship between any two VCs may represent a different syndication. Different relationships in the heterogeneous networks must be measured to obtain precious information and enhance community detection. This structural information may be particularly important when one wants to obtain more concrete and accurate results. Likewise, this paper does not consider other community-related methods besides clustering. Third, BiPClust is an extensive method of distance-based clustering that relies on the choice of distance metric used to calculate similarities between data points. Different distance metrics may lead to different clustering results, and selecting an appropriate distance metric can be challenging. Moreover, it assumes that the data is continuous and that clusters have a spherical shape and similar size, which may not be true for all datasets. In addition, the regularized k-means clustering in the current paper is based on a combination of group lasso penalty and fused lasso penalty, but can be further extended to other penalty functions and adjusted according to one's needs [39]. Further discussion on regularized clustering problems with different penalty functions is required.

One of the main shortcomings of distance-based clustering is that it heavily relies on the selection of distance metric used to calculate the similarity between data points. Different distance metrics may lead to different clustering results, and selecting an appropriate distance metric can be challenging, particularly when dealing with high-dimensional or complex data. Additionally, distance-based clustering assumes that the data is continuous and that clusters have a spherical shape and similar size, which may not be true for all datasets. Finally, distance-based clustering methods are sensitive to outliers and noise in the data, which can considerably affect the clustering results.

Our study has positive implications for international management research, the practice of international VC investing, and public policies that aim to stimulate venture-driven ecosystems. We propose a new community detection method and rigorously verify the reliability of this method through theoretical proof and experiment. Notably, we consider both the attribute information and the network structure, which is more in line with most networks in the real world (nodes in these networks are usually not mere points but have richer features). We validate our approach in a practical sociological scenario, investigating the community structure of Chinese VC networks over three time periods and obtaining some inspiring discoveries, which we plan to investigate further in our future study. Community detection can be a widely used technique in various fields, including social network analysis, biology, computer science, transportation, and finance. For example, community detection can be used to identify clusters of related financial assets or market participants in financial networks, such as stock markets, credit networks, and payment systems. This information can be used to understand market dynamics, identify systemic risks, and design optimal investment strategies. Hence, our study may encourage further research in community detection and social networks as well as at the intersection between them.

CRediT authorship contribution statement

Hu Yang: Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Writing – original draft, Writing – review & editing. **Wenjing Xiang:** Data curation, Formal analysis, Investigation, Resources, Software, Validation, Visu-

alization, Writing – original draft. **Jar-Der Luo:** Data curation, Supervision. **Qiuyan Zhang:** Formal analysis, Funding acquisition, Investigation, Writing – original draft.

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.

Data availability

The authors do not have permission to share data.

Acknowledgements

H.Y. was supported by the NSSF 22FGLB056, the National Statistical Science Foundation of China Project number 2023LY078, Program for innovation Research in CUFE, and the Emerging Interdisciplinary Project of CUFE. Q.Y. Zhang was partially supported by NSFC 12201430, 11971097, and Capital University of Economics and Business: The Fundamental Research Funds for Beijing Universities XRZ2021044.

Appendix A. Proof of Theorem 1

Proof 1. There exists a closed ball $B(M)$, which is centered at the origin with radius M and contains all the estimated cluster centers when n is large enough. Thus, $B(M)$ contains the estimated cluster centers $\hat{\mu}_{1\cdot},...,\hat{\mu}_{K\cdot}$. According to the analysis in Pollard [28], there exists a ball $B(\tilde{M})$, which is centered at the origin with radius $\tilde{M}$ and contains the estimated cluster centers $\{\tilde{\mu}_{1\cdot},...,\tilde{\mu}_{K\cdot}\}$, when n is large enough. Then, we conclude that $B(\tilde{M}) \subset B(M)$. The optimization problem (17) can be equivalently transformed into

$$W(U,P_n)=\int \min_{\mu_{k\cdot}\in U}||x-\mu_{k\cdot}||_2^2 P_n(dx), \quad s.t. \quad \sum_{j=1}^{p}\left\{\sum_{k=2}^{K}\dot{\psi}\left(D^k\mu_{\cdot j}\right)+\dot{\varphi}(\mu_{\cdot j})\right\}\le s_n, \tag{A.0}$$

where s_n is a sequence, and $s_n \to \infty$ as $n \to \infty$. For a sufficiently large s_N,

$$\left\{U : \sum_{j=1}^{p}\left\{\sum_{k=2}^{K}\dot{\psi}\left(D^k\mu_{\cdot j}\right)+\dot{\varphi}(\mu_{\cdot j})\right\}\le s_N\right\} \supset B(\tilde{M}) \tag{A.1}$$

Thus, $B(M)=B(\tilde{M})$ contains $\{\hat{\mu}_{1\cdot},...,\hat{\mu}_{K\cdot}\}$. It achieves the unique minimum, which lies in a compact set of $\mathbf{R}^p$.

Note that

$$\begin{aligned}
&\sup_{U\in B(M)}|W(U,P_n)-W(U,P)| \\
\le &\sup_{U\in B(M)}\left|\int \min_{\mu_{k\cdot}\in U}||x-\mu_{k\cdot}||_2^2 P_n(dx)-\int \min_{\mu_{k\cdot}\in U}||x-\mu_{k\cdot}||_2^2 P(dx)\right| \\
&+\sup_{U\in B(M)}\sum_{j=1}^{p}\left\{\sum_{k=1}^{K}\dot{\psi}\left(D^k\mu_{\cdot j}\right)+\dot{\varphi}(\mu_{\cdot j})\right\}.
\end{aligned} \tag{A.2}$$

Based on the strong law of large number

$$\sup_{U\in B(M)}\left|\int \min_{\mu_{k\cdot}\in U}||x-\mu_{k\cdot}||_2^2 P_n(dx)-\int \min_{\mu_{k\cdot}\in U}||x-\mu_{k\cdot}||_2^2 P(dx)\right| \tag{A.3}$$

almost surely converges to zero. Note that

$$\begin{aligned}
||D^k\mu_{\cdot j}||_1 &= \sum_{i=1}^{k-1}||\mu_{ij}-\mu_{kj}||_1 \\
&\le \sum_{i=1}^{k-1}|\mu_{ij}|+(k-1)|\mu_{kj}| \\
&\le (k-1)||\mu_{\cdot j}||_1.
\end{aligned} \tag{A.4}$$

Applying the inequality $\frac{\sum_{i=1}^n x_i}{n}\le\sqrt{\frac{\sum_{i=1}^n x_i^2}{n}}$, we have

$$||D^k\mu_{\cdot j}||_1 \le (k-1)^{3/2}||\mu_{\cdot j}||_2. \tag{A.5}$$

Thus,

$$\frac{||D^k\mu_{\cdot j}||_1}{||\tilde{\mu}_{\cdot j}||_2} \leq \frac{(k-1)^{3/2}||\mu_{\cdot j}||_2}{||\tilde{\mu}_{\cdot j}||_2}. \tag{A.6}$$

By the boundedness of $||\mu_{\cdot j}||_2$, and $||\hat{\mu}_{\cdot j}||_2$ over $B(M)$, and the condition $n^{1/2}\lambda_l p \to 0$, for $l = 1, 2$, one obtains that $W(U, P_n)$ converges almost surely to $W(U, P)$ uniformly over the subsets of $B(M)$.

Appendix B. Proof of Theorem 2

Proof 2. If $\hat{\mu}_{\cdot p} \neq 0$, the derivative with respect to the components of $||\hat{\mu}_{\cdot p}||_2$ can be derived. According to the Karush-Kuhn-Tucker condition, we have

$$\begin{aligned}
0 &= -\frac{2}{\sqrt{n}}\hat{L}^T(Z_{\cdot p} - \hat{L}\hat{\mu}_{\cdot p}) + \sqrt{n}\lambda_1\frac{1}{||\tilde{\mu}_{\cdot p}||_2}\sum_{k=1}^{K} sign(\hat{\mu}_{\cdot p} - \hat{\mu}_{kp}\mathbf{1}'_K) + \sqrt{n}\lambda_2\frac{\hat{\mu}_{\cdot p}}{||\hat{\mu}_{\cdot p}||_2||\tilde{\mu}_{\cdot p}||_2} \\
&= \frac{2}{\sqrt{n}}(\hat{L}^T - \bar{L}^T)\bar{L}\hat{\mu}_{\cdot p} - \frac{2}{\sqrt{n}}(\hat{L}^T - \bar{L}^T)\epsilon_{\cdot p} + \frac{2}{\sqrt{n}}(\hat{L}^T - \bar{L}^T)(\hat{L} - \bar{L})\hat{\mu}_{\cdot p} \\
&\quad + \frac{2}{\sqrt{n}}\bar{L}^T(\hat{L} - \bar{L})\hat{\mu}_{\cdot p} - \frac{2}{\sqrt{n}}\bar{L}^T\epsilon_{\cdot p} + \left(\frac{2}{n}\bar{L}^T\bar{L} + \lambda_2\frac{1}{||\hat{\mu}_{\cdot p}||_2||\tilde{\mu}_{\cdot p}||_2}\right)\sqrt{n}\hat{\mu}_{\cdot p} \\
&\quad + \sqrt{n}\lambda_1\frac{1}{||\tilde{\mu}_{\cdot p}||_2}\sum_{k=1}^{K} sign(\hat{\mu}_{\cdot p} - \hat{\mu}_{kp}\mathbf{1}'_K).
\end{aligned} \tag{A.7}$$

Based on the fact that $\hat{L}$ converges in probability to $\bar{L}$, we only need to calculate the order of last two terms. Here,

$$\left|\left|\left(\frac{2}{n}\bar{L}^T\bar{L} + \lambda_2\frac{1}{||\hat{\mu}_{\cdot p}||_2||\tilde{\mu}_{\cdot p}||_2}\right)\sqrt{n}\hat{\mu}_{\cdot p} + \sqrt{n}\lambda_1\frac{1}{||\tilde{\mu}_{\cdot p}||_2}\sum_{k=1}^{K} sign(\hat{\mu}_{\cdot p} - \hat{\mu}_{kp}\mathbf{1}'_K)\right|\right|_2 \tag{A.8}$$

$$\geq \left|\left|\lambda_2\frac{1}{||\hat{\mu}_{\cdot p}||_2||\tilde{\mu}_{\cdot p}||_2}\sqrt{n}\hat{\mu}_{\cdot p} + \sqrt{n}\lambda_1\frac{1}{||\tilde{\mu}_{\cdot p}||_2}\sum_{k=1}^{K} sign(\hat{\mu}_{\cdot p} - \hat{\mu}_{kp}\mathbf{1}'_K)\right|\right|_2 \tag{A.9}$$

$$= \left|\left|\frac{\lambda_1\sqrt{n}}{||\tilde{\mu}_{\cdot p}||_2}\left(\frac{1}{||\hat{\mu}_{\cdot p}||_2}\hat{\mu}_{\cdot p} + \sum_{k=1}^{K} sign(\hat{\mu}_{\cdot p} - \hat{\mu}_{kp}\mathbf{1}'_K)\right)\right|\right|_2. \tag{A.10}$$

Let $v = \sum_{k=1}^{K} sign(\hat{\mu}_{\cdot p} - \hat{\mu}_{kp}\mathbf{1}'_K)$, then

$$(A.8) \geq \frac{\lambda_1\sqrt{n}}{||\tilde{\mu}_{\cdot p}||_2||\hat{\mu}_{\cdot p}||_2}\left|\left|\hat{\mu}_{\cdot p} + ||\hat{\mu}_{\cdot p}||_2 v\right|\right|_2. \tag{A.11}$$

Note that

$$0 < \frac{\left|\left|\hat{\mu}_{\cdot p} + ||\hat{\mu}_{\cdot p}||_2 v\right|\right|_2}{||\hat{\mu}_{\cdot p}||_2} \leq \frac{\left|\left|\hat{\mu}_{\cdot p}\right|\right|_2 + ||\hat{\mu}_{\cdot p}||_2||v||_2}{||\hat{\mu}_{\cdot p}||_2} = 1 + ||v||_2, \tag{A.12}$$

where $||v||_2$ is a constant order quantity. Consequently, $\left|\left|\hat{\mu}_{\cdot p} + ||\hat{\mu}_{\cdot p}||_2 v\right|\right|_2$ and $||\hat{\mu}_{\cdot p}||_2$ have same order. Given that

$$\frac{\lambda_1\sqrt{n}}{||\tilde{\mu}_{\cdot p}||_2||\hat{\mu}_{\cdot p}||_2}||\hat{\mu}_{\cdot p}||_2 = O(n\lambda_l), \tag{A.13}$$

and the assumption $n^{-2}\lambda_l^{-2}p \to 0$, we conclude that (A.8) goes to infinity, which leads to a contradiction. Thus, $||\hat{\mu}_{\cdot p}||_2$ is 0 with probability 1.

Appendix C. Remark of Algorithm 1

Remark 1. Strictly speaking, after algebraic calculation, we derive

$$\mu_{\cdot j} = \left(L^T L + \lambda_1\sum_{k=1}^{K}(D^k)^T D^k + \frac{\lambda_2 w_j}{||\mu_{\cdot j}||_2^2}\mathbf{I}\right)^{-1} \cdot \left(L^T Z_{\cdot j} + \lambda_1\sum_{k=1}^{K}(D^k)^T\Theta^k_{\cdot j}\right). \tag{A.14}$$

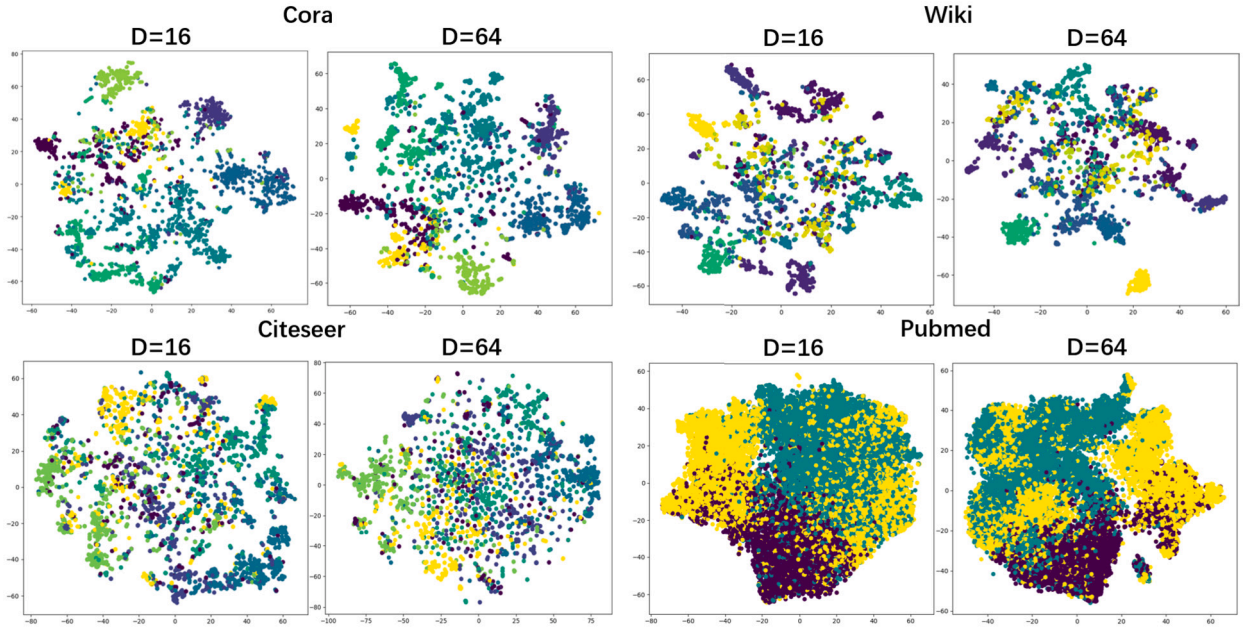

Fig. A.1. t-SNE plot visualizing cluster assignments of actors in two benchmark datasets.

According to the properties of the L_2 norm and the formula $(B+XY^T)^{-1} = B^{-1} - \frac{B^{-1}XY^TB^{-1}}{1+Y^TB^{-1}X}$, one can conclude that $||\mu_{\cdot j}||_2$ is bounded, i.e.

$$||\dot{\mu}_{\cdot j}||_2 - \left|\left|\frac{(D_{diag})^{-1}\mathbf{1}\mathbf{1}^T(D_{diag})^{-1}\left(L^TZ_{\cdot j} + \lambda_1\sum_{k=1}^{K}(D^k)^T\Theta^k_{\cdot j}\right)}{1-\mathbf{1}^T(D_{diag})^{-1}\mathbf{1}}\right|\right|_2 \leq ||\mu_{\cdot j}||_2, \tag{A.15}$$

and

$$\leq ||\mu_{\cdot j}||_2 \leq ||\dot{\mu}_{\cdot j}||_2 + \left|\left|\frac{(D_{diag})^{-1}\mathbf{1}\mathbf{1}^T(D_{diag})^{-1}\left(L^TZ_{\cdot j} + \lambda_1\sum_{k=1}^{K}(D^k)^T\Theta^k_{\cdot j}\right)}{1-\mathbf{1}^T(D_{diag})^{-1}\mathbf{1}}\right|\right|_2, \tag{A.16}$$

where $D_{diag} = L^TL + \lambda_1 K\mathbf{I} + \frac{\lambda_2 w_j}{||\mu_{\cdot j}||_2^2}\mathbf{I}$, and $\dot{\mu}_{\cdot j} = D_{diag}^{-1}\cdot\left(L^TZ_{\cdot j} + \lambda_1\sum_{k=1}^{K}(D^k)^T\Theta^k_{\cdot j}\right)$. Under appropriate conditions, we get

$$\left|\left|(D_{diag})^{-1}\mathbf{1}\mathbf{1}^T(D_{diag})^{-1}\left(L^TZ_{\cdot j} + \lambda_1\sum_{k=1}^{K}(D^k)^T\Theta^k_{\cdot j}\right)\right|\right|_2 \tag{A.17}$$

to converge to zero in probability. Hence why we update $\mu^{(s+1)}_{\cdot j}$ by (27).

Appendix D. t-SNE plot visualizing cluster assignments of actors in two benchmark datasets

We visualize the samples by using t-Distributed Stochastic Neighbor Embedding (t-SNE) (a technique for dimensionality reduction that is particularly well-suited for the visualization of high-dimensional datasets) to show whether the embedding represents the actors well when we know the true label. Fig. A.1 shows that the boundaries between the clusters are not completely clear. It probably contains noise information for distinguishing samples that needs to be removed during clustering.

Appendix E. Evaluation of clustering results based on different embedding methods with fixed dimension

We perform community detection on different embedding obtained by different VGAE-based methods (such as ARGA, ARVGA, LMAVGAE, GMAVGAE, DGAE and DGVAE) on four benchmark datasets, helping us gain insight into its strengths and weaknesses and determining whether it is a viable option for clustering tasks where the number of clusters is not known in advance. Moreover, this experiment helps us understand how the proposed method performs with the unknown number of clusters. Because k-means clustering algorithm has been validated performing well on detecting communities in previous studies [12], we use it as a cluster. Considering it cannot overcome the influence of useless latent variables, we fix the dimension of embedding space to a predefined

Table A.1
Evaluation of clustering results based on different embedding methods with fixed dimension.

Datasets	Method	# of Clusters	ACC	PURITY	F1	NMI	ARI
Cora	VGAE+BiClust	non fixed	65.30%	63.50%	48.13%	43.65%	37.35%
	LMAVGAE+kmeans	fixed	66.78%	68.63%	53.23%	49.59%	42.85%
	GMAVGAE+kmeans	fixed	58.91%	61.49%	44.97%	42.40%	33.30%
	ARVGE+kmeans	fixed	62.77%	65.29%	47.76%	45.01%	37.12%
	ARGE+kmeans	fixed	64.01%	65.59%	50.58%	44.13%	40.23%
	DGVAE+kmeans	fixed	60.19%	62.70%	45.97%	44.45%	37.01%
	DGAE+kmeans	fixed	62.00%	62.00%	46.96%	47.09%	34.51%
	Node2V+kmeans	fixed	61.96%	63.52%	47.06%	42.09%	35.52%
	DW+kmeans	fixed	39.33%	43.32%	31.51%	22.10%	10.37%
Citeseer	VGAE+BiClust	non fixed	44.57%	52.21%	31.61%	22.61%	16.23%
	LMAVGAE+kmeans	fixed	43.75%	47.81%	31.37%	22.01%	14.59%
	GMAVGAE+kmeans	fixed	44.24%	47.22%	31.24%	19.24%	14.81%
	ARVGE+kmeans	fixed	40.67%	44.75%	29.43%	18.80%	13.88%
	ARGE+kmeans	fixed	42.82%	47.16%	29.54%	19.42%	13.04%
	DGVAE+kmeans	fixed	40.09%	41.33%	28.20%	17.11%	11.40%
	DGAE+kmeans	fixed	41.11%	45.61%	30.43%	19.76%	13.75%
	Node2V+kmeans	fixed	42.05%	44.72%	30.12%	15.34%	14.18%
	DW+kmeans	fixed	36.61%	37.27%	29.97%	10.72%	11.99%
Wiki	VGAE+BiClust	non fixed	45.47%	46.61%	32.83%	40.05%	24.92%
	LMAVGAE+kmeans	fixed	45.01%	56.41%	32.08%	44.51%	26.18%
	GMAVGAE+kmeans	fixed	28.94%	38.85%	20.28%	26.46%	11.95%
	ARVGE+kmeans	fixed	40.72%	53.64%	28.48%	41.01%	22.53%
	ARGE+kmeans	fixed	44.96%	58.94%	31.97%	45.13%	25.99%
	DGVAE+kmeans	fixed	41.12%	52.94%	27.91%	38.57%	22.28%
	DGAE+kmeans	fixed	25.82%	37.34%	14.98%	11.41%	4.27%
	Node2V+kmeans	fixed	41.87%	53.18%	28.90%	38.50%	23.01%
	DeepW+kmeans	fixed	42.28%	52.10%	30.20%	39.22%	22.60%
Pubmed	VGAE+BiClust	non fixed	66.65%	66.65%	51.92%	26.05%	26.01%
	LMAVGAE+kmeans	fixed	65.42%	65.42%	52.02%	25.91%	25.41%
	GMAVGAE+kmeans	fixed	66.82%	66.82%	53.12%	26.96%	27.24%
	ARVGE+kmeans	fixed	65.59%	65.59%	51.13%	24.90%	24.69%
	ARGE+kmeans	fixed	66.13%	66.13%	52.66%	25.83%	26.17%
	DGVAE+kmeans	fixed	65.81%	65.81%	51.66%	26.16%	25.53%
	DGAE+kmeans	fixed	67.08%	67.08%	53.41%	26.99%	27.60%
	Node2v+kmeans	fixed	68.82%	68.82%	54.99%	27.49%	30.33%
	DeepW+kmeans	fixed	66.37%	66.37%	53.69%	26.57%	28.73%

number 16 which is the same number as previous studies. Results in Table A.1 indicate the proposed method can find the suitable combination of hyperparameters to optimize CH and DBI scores.

Appendix F. Brief description of VCs' 28 attributes derived from investment events

Besides syndication networks, VCs' attributes derived from investment events are also considered in our analysis. Each event indicates that a VC firm has invested in a startup. Investment information includes which startup a VC has invested in, at what time, and in which place, and also lists which industry the startup belongs to, in which investing period (such as initial stage, expansion stage, seed stage) it is at, and in which investing round (A, B, C, D, E, F and G round) it is at. Usually, most investors prefer to invest in several firms, industries, areas, stages and rounds but not one, because they want to diversify investment risks. Some VCs have more resources than others to do so. They tend to operate on a larger scale and have greater experience than those who can only invest small amounts. Therefore, such attributes related to scale and experience are vital to describe VCs, including 28 other features. A brief description of them is presented in Table A.2.

Appendix G. Density of the communities

We calculate the density of each community, defined as $Den = \frac{2*|E|}{n*(n-1)}$, where $|E|$ is the number of edges and n is the number of nodes in a community, also presented in Table A.3. We denote the dominant communities in three networks from 2001 to 2015 as $\{C_1,C_2,...,C_9\}$ and the remaining ones as $\{O_1,O_2,...O_8\}$. Specifically, C_1, C_2 and C_3 are the three most connected communities in the syndicated investment network for the first period (2001-2005). Similarly, C_4, C_5 and C_6 dominate the network in the second period (2006-2010), while C_7, C_8 and C_9 are the most important ones in the latest period (2011-2015). Despite the increasing scale and connections of the communities, their density decrease over time because the increasing speed of edges is slower than that of nodes.

Table A.2
Brief description of VCs' 28 attributes derived from investment events.

Features	Abbreviation	Description
X1	Age	When the VC was founded
X2	Investments	The cumulative # of investments since 2001
X3	Firms	The cumulative # of investing distinct startups
X4	Nations	The cumulative # of investing distinct countries
X5	Province	The cumulative # of investing distinct provinces
X6	YRD	The cumulative # of investing industries in Yangtze River Delta
X7	PRD	The cumulative # of investing industries in Pearl River Delta
X8	BTH	The cumulative # of investing industries in Beijing-Tianjin-Hebei
X9	Other_Area	The cumulative # of investing industries in Other areas
X10	Industries	The cumulative # of investing distinct industries
X11	Computer	The cumulative # of investing startups in computer science
X12	Medicine	The cumulative # of investing startups in medicine, healthcare and life sciences
X13	Biotech	The cumulative # of investing startups in Biotechnology
X14	Semicon	The cumulative # of investing startups in semiconductor and electronic equipment
X15	Commu	The cumulative # of investing startups in communication, culture and entertainment
X16	Other_ind	The cumulative # of investing startups in other fields
X17	Stages	The cumulative # of investing stages
X18	Seed	The cumulative # of investing startups at seed stage
X19	Init	The cumulative # of investing startups at initial stage
X20	Expansion	The cumulative # of investing startups at expansion stage
X21	Rounds	The cumulative # of investing rounds
X22	A	The cumulative # of investing startups in A round
X23	B	The cumulative # of investing startups in B round
X24	C	The cumulative # of investing startups in C round
X25	D	The cumulative # of investing startups in D round
X26	E	The cumulative # of investing startups in E round
X27	F	The cumulative # of investing startups in F round
X28	G	The cumulative # of investing startups in G round

Table A.3
Density of the communities.

2001-2005	C1	C2	C3	O1	O2	O3
	0.2092	0.1477	0.1228	0.0444	0.0654	0.0688
2006-2010	C4	C5	C6	O4	O5	O6
	0.0655	0.0386	0.0461	0.0200	0.0253	0.0150
2011-2015	C7	C8	C9	O7	O8	
	0.0430	0.0140	0.0452	0.0097	0.0045	

Appendix H. Clustering analysis of data with different numbers of embedded features

By comparing the clustering results for different numbers of embeddings except 16 and 64, we demonstrate the impact that dimensionality has on the quality of clustering results and the importance of selecting an appropriate number of embeddings for a given dataset. VGAE-based methods are applied to learn the representation of Cora given a dimension of embedding (e.g., 24, 32 and 48) first, and then various clustering methods have performed on Cora and obtained the clustering results to compare the clustering results for different numbers of embedding. Results in Tables A.4 and A.5 show the number of embedded features or the dimensionality of the embedding space can have a significant impact on the clustering performance. However, the proposed new method outperforms other methods regardless of the number of embedded features set, indicating that it is a robust and effective method.

Appendix I. Clustering analysis of data with different tuning parameters

To illustrate the process of tuning parameters, we perform the proposed clustering method on the Cora dataset with different tuning parameters and evaluate the clustering results using same metrics mentioned before previous section. Our goals are searching the optimal parameters (including λ_1, λ_2 and λ_3) to maximize CH and DBI indexes using the strategy of a grid search. First, we define a grid of hyperparameters to search over. Second, for each combination of hyperparameters, we apply the clustering algorithm to our dataset and compute the CH and DBI indices. Then, we store the results of each combination of hyperparameters, along with the corresponding CH and DBI scores. Next, we analyze the results and identify the hyperparameters that maximize the CH and DBI indices. Finally, we select the optimal hyperparameters that produce the highest CH and lowest DBI scores and use them to perform clustering on the full dataset. Results in Table A.6 indicate the proposed method can find the suitable combination of hyperparameters to optimize CH and DBI scores.

Table A.4
Evaluation of clustering results of Cora based on VGAE with different embedding dimensions.

# of Embedding	Method	# of Clusters	Purity	Accuracy	F1	ARI	NMI
16	BiPClust	7	65.30%	63.50%	48.13%	37.35%	43.65%
	Kmeans	7.8	65.10%	57.13%	44.13%	33.90%	42.89%
	HClust	7.5	61.86%	57.98%	45.39%	33.71%	40.38%
	Sparcl	8	62.25%	55.51%	41.22%	30.46%	39.85%
	Gclust	9	66.17%	55.23%	42.36%	32.86%	42.51%
	PClust	8.75	64.93%	56.28%	43.07%	33.06%	43.03%
24	BiPClust	7.25	65.24%	62.69%	49.58%	37.39%	44.39%
	Kmeans	8.25	63.66%	57.04%	43.46%	32.26%	41.87%
	HClust	9	64.95%	56.27%	42.50%	30.99%	40.76%
	Sparcl	8.25	57.27%	49.92%	36.05%	23.94%	34.25%
	Gclust	9.75	66.23%	51.08%	41.10%	31.39%	42.13%
	PClust	6.75	58.35%	52.37%	42.66%	26.56%	38.90%
32	BiPClust	7.75	64.38%	61.52%	47.84%	35.21%	43.88%
	Kmeans	7	61.14%	56.59%	44.09%	30.42%	41.70%
	HClust	10.5	62.94%	49.54%	38.11%	25.50%	39.63%
	Sparcl	7	53.24%	48.92%	35.50%	21.65%	30.67%
	Gclust	8.75	65.43%	52.41%	42.01%	31.20%	41.34%
	PClust	8.5	64.74%	55.39%	44.64%	31.33%	43.81%
48	BiPClust	8	65.07%	58.96%	49.17%	35.87%	45.13%
	Kmeans	6	56.98%	54.40%	43.28%	26.66%	38.65%
	HClust	10.25	61.14%	46.74%	36.50%	21.03%	38.02%
	Sparcl	6	45.97%	41.08%	30.66%	13.92%	23.69%
	Gclust	6.5	59.82%	56.51%	46.48%	33.06%	41.67%
	PClust	6.75	58.35%	53.64%	44.38%	26.39%	40.58%
64	BiPClust	9.6	66.26%	56.80%	44.39%	31.22%	43.64%
	Kmeans	5.6	52.84%	51.11%	38.93%	21.81%	36.52%
	HClust	12.2	61.52%	44.10%	33.42%	18.10%	37.43%
	Sparcl	5.75	43.57%	41.62%	29.82%	13.30%	20.58%
	Gclust	6	60.25%	54.94%	44.46%	28.69%	40.77%
	PClust	7.5	60.08%	54.34%	43.11%	24.30%	42.68%

Table A.5
Evaluation of clustering results of Cora based on LMAVGAE with different embedding dimensions.

# of Embedding	Method	# of Clusters	Purity	Accuracy	F1	ARI	NMI
16	BiPClust	6.8	69.43%	66.27%	53.88%	43.22%	50.45%
	Kmeans	7.8	68.21%	58.68%	47.97%	37.47%	46.90%
	HClust	7.6	66.49%	62.94%	49.52%	37.73%	45.44%
	Sparcl	7.8	58.80%	50.64%	38.00%	26.14%	36.72%
	Gclust	9	71.62%	57.84%	47.07%	37.83%	48.84%
	PClust	8.6	67.61%	58.09%	46.01%	35.57%	46.41%
24	BiPClust	7.75	70.51%	67.51%	54.28%	42.75%	49.94%
	Kmeans	7.25	67.02%	62.63%	50.63%	39.09%	47.07%
	HClust	8	65.27%	59.69%	45.81%	32.32%	43.57%
	Sparcl	7.25	52.08%	44.15%	24.29%	18.92%	27.35%
	Gclust	6.25	62.32%	60.10%	48.62%	36.30%	44.64%
	PClust	6	63.09%	63.09%	48.59%	32.70%	46.54%
32	BiPClust	8	68.40%	63.66%	50.92%	38.50%	48.12%
	Kmeans	6.4	61.11%	58.85%	44.59%	29.60%	45.37%
	HClust	10	63.72%	52.51%	41.07%	26.14%	41.37%
	Sparcl	6.4	48.84%	45.37%	32.69%	18.20%	24.68%
	Gclust	7.2	64.76%	57.45%	45.01%	32.75%	43.90%
	PClust	8.2	66.60%	60.13%	47.35%	33.17%	47.31%
48	BiPClust	6.67	65.79%	63.32%	47.29%	31.27%	46.02%
	Kmeans	6.67	59.77%	56.67%	43.20%	27.10%	41.78%
	HClust	12.33	67.90%	46.22%	36.18%	24.24%	42.92%
	Sparcl	6.67	41.75%	34.27%	25.24%	9.72%	15.78%
	Gclust	7.67	65.26%	59.64%	45.67%	32.85%	44.58%
	PClust	7.33	64.45%	59.26%	43.99%	25.65%	44.42%
64	BiPClust	9	65.76%	57.21%	45.21%	31.14%	45.30%
	Kmeans	6.4	56.96%	55.66%	42.70%	24.62%	37.53%

(*continued on next page*)

Table A.5 (*continued*)

# of Embedding	Method	# of Clusters	Purity	Accuracy	F1	ARI	NMI
	HClust	16.2	63.83%	40.36%	31.52%	16.81%	38.71%
	Sparcl	6.4	37.68%	31.93%	23.84%	7.57%	12.26%
	Gclust	8	61.81%	52.58%	40.04%	26.41%	39.83%
	PClust	7.4	60.33%	54.31%	41.27%	21.33%	42.09%

Table A.6
Evaluation of clustering results of Cora based on VGAE with different embedding dimensions.

λ_1	λ_2	λ_3	CH	DBI	# of Clusters	Purity	Accuracy	F1	ARI	NMI
0.5	0.375	3	43.33	459.54	14	67.02%	43.02%	33.63%	20.79%	42.24%
		5	35.40	1014.86	2	30.83%	30.83%	30.25%	4.15%	6.30%
		7	46.87	512.52	9	70.49%	63.44%	51.14%	39.38%	46.33%
	0.4	3	43.86	466.64	13	62.15%	41.80%	32.31%	18.68%	39.03%
		5	46.93	513.95	7	57.79%	55.35%	45.00%	27.87%	41.94%
		7	44.51	929.18	2	39.51%	39.51%	36.66%	13.28%	17.33%
	0.425	3	44.13	403.06	8	56.31%	45.86%	39.13%	17.21%	38.41%
		5	48.95	547.71	5	49.96%	45.46%	37.69%	13.97%	33.71%
		7	43.27	646.59	5	50.15%	46.05%	41.73%	23.83%	29.68%
1	0.375	3	42.47	465.32	16	70.57%	41.58%	33.44%	23.79%	43.04%
		5	45.62	489.54	11	69.83%	57.79%	47.26%	35.53%	46.06%
		7	0.00	0.00	1	30.21%	30.21%	30.40%	0.00%	0.00%
	0.4	3	46.40	463.11	14	70.24%	43.13%	33.83%	22.56%	43.52%
		5	48.06	513.37	8	57.13%	45.64%	37.86%	20.51%	39.00%
		7	46.86	806.99	2	39.07%	39.07%	33.67%	7.89%	14.98%
	0.425	3	47.91	473.11	10	69.98%	59.38%	49.78%	36.80%	47.52%
		5	46.04	497.29	12	70.86%	48.97%	39.22%	28.19%	43.80%
		7	0.00	0.00	1	30.21%	30.21%	30.40%	0.00%	0.00%

References

[1] S. Agrawal, A. Patel, Sag cluster: an unsupervised graph clustering based on collaborative similarity for community detection in complex networks, Phys. A, Stat. Mech. Appl. 563 (2021) 125459.
[2] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
[3] Y. Chen, D. Mo, Community detection for multilayer weighted networks, Inf. Sci. 595 (2022) 119–141.
[4] P. Chunaev, Community detection in node-attributed social networks: a survey, Comput. Sci. Rev. 37 (2020) 100286.
[5] F. Chung, F. Graham, Eigenvalues and the Laplacian of a Graph, American Mathematical Society, 1997.
[6] G. Cui, J. Zhou, C. Yang, Z. Liu, Adaptive graph encoder for attributed graph embedding, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 976–985.
[7] P. Cui, X. Wang, J. Pei, W. Zhu, A survey on network embedding, IEEE Trans. Knowl. Data Eng. 31 (2018) 833–852.
[8] S. Fortunato, D. Hric, Community detection in networks: a user guide, Phys. Rep. 659 (2016) 1–44.
[9] J.H. Friedman, J.J. Meulman, Clustering objects on subsets of attributes (with discussion), J. R. Stat. Soc., Ser. B, Stat. Methodol. 66 (2004) 815–849.
[10] H. Gao, H. Huang, Deep attributed network embedding, in: Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI), 2018.
[11] W. Gu, A. Tandon, Y.Y. Ahn, F. Radicchi, Principled approach to the selection of the embedding dimension of networks, Nat. Commun. 12 (2021) 1–10.
[12] S.G. Guillaume, J.F. Lutzeyer, G. Dasoulas, R. Hennequin, M. Vazirgiannis, Modularity-aware graph autoencoders for joint community detection and link prediction, Neural Netw. 153 (2022) 474–495.
[13] Y.V. Hochberg, Whom you know matters: venture capital networks and investment performance, J. Finance 62 (2007) 251–301.
[14] Z. Huang, Y. Wang, X. Ma, Clustering of cancer attributed networks by dynamically and jointly factorizing multi-layer graphs, IEEE/ACM Trans. Comput. Biol. Bioinform. 19 (2021) 2737–2748.
[15] A.M. Ikotun, A.E. Ezugwu, L. Abualigah, B. Abuhaija, J. Heming, K-means clustering algorithms: a comprehensive review, variants analysis, and advances in the era of big data, Inf. Sci. (2022).
[16] D. Jin, Z. Yu, P. Jiao, S. Pan, P.S. Yu, W. Zhang, A survey of community detection approaches: from statistical modeling to deep learning, IEEE Trans. Knowl. Data Eng. (2021), https://doi.org/10.1109/TKDE.2021.3104155.
[17] Y. Jin, Q. Zhang, S.P. Li, Topological properties and community detection of venture capital network: evidence from China, Phys. A, Stat. Mech. Appl. 442 (2016) 300–311.
[18] B. Jing, C. Park, H. Tong, Hdmi: high-order deep multiplex infomax, in: Proceedings of the Web Conference 2021, 2021, pp. 2414–2424.
[19] J. Kim, J.G. Lee, Community detection in multi-layer graphs: a survey, SIGMOD Rec. 44 (2015) 37–48.
[20] T. Kipf, M. Welling, Variational graph auto-encoders, ArXiv, arXiv:1611.07308 [abs], 2016.
[21] H. Li, D. Pati, Variable selection using shrinkage priors, Comput. Stat. Data Anal. 107 (2017) 107–119.
[22] Y. Lu, Y.m.Cheung, Y. Tang, Self-adaptive multiprototype-based competitive learning approach: a k-means-type algorithm for imbalanced data clustering, IEEE Trans. Cybern. (2019) 1–15.
[23] S. Ma, J. Huang, Penalized feature selection and classification in bioinformatics, Brief. Bioinform. 9 (2008) 392–403.
[24] L.v.d. Maaten, G.E. Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (2008) 2579–2605.
[25] W. Pan, X. Shen, B. Liu, Cluster analysis: unsupervised learning via supervised learning with a non-convex penalty, J. Mach. Learn. Res. 14 (2013).
[26] C. Park, D. Kim, J. Han, H. Yu, Unsupervised attributed multiplex network embedding, Proc. AAAI Conf. Artif. Intell. 34 (2020) 5371–5378.
[27] P. Pham, L.T. Nguyen, W. Pedrycz, B. Vo, Deep learning, graph-based text representation and classification: a survey, perspectives and challenges, Artif. Intell. Rev. (2022) 1–35.

[28] D. Pollard, A central limit theorem for k-means clustering, Ann. Probab. 10 (1982) 919–926.
[29] M.J. Rattigan, M.E. Maier, D.D. Jensen, Graph clustering with network structure indices, in: Proceedings of the 24th International Conference on Machine Learning, 2007, pp. 783–790.
[30] S.A. Rice, The identification of blocs in small political bodies, Am. Polit. Sci. Rev. 21 (1927) 619–627.
[31] X. Shen, F.l. Chung, Deep network embedding for graph representation learning in signed networks, IEEE Trans. Cybern. 50 (2018) 1556–1568.
[32] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of go with deep neural networks and tree search, Nature 529 (2016) 484–489.
[33] W. Sun, J. Wang, Y. Fang, Regularized k-means clustering of high-dimensional data and its asymptotic consistency, Electron. J. Stat. 6 (2012) 148–167.
[34] R. Tibshirani, Regression shrinkage and selection via the lasso, J. R. Stat. Soc., Ser. B, Methodol. 58 (1996) 267–288.
[35] U. Von Luxburg, et al., Clustering stability: an overview, Found. Trends Mach. Learn. 2 (2010) 235–274.
[36] R. Wilson, The theory of syndicates, Econometrica 36 (1968) 119–132.
[37] D.M. Witten, R. Tibshirani, A framework for feature selection in clustering, J. Am. Stat. Assoc. 105 (2010) 713–726.
[38] Y. Wu, Y. Fu, J. Xu, H. Yin, Q. Zhou, D. Liu, Heterogeneous question answering community detection based on graph neural network, Inf. Sci. 621 (2023) 652–671.
[39] D. Xia, Y. Yang, S. Yang, T. Li, Incomplete multi-view clustering via kernelized graph learning, Inf. Sci. 625 (2023) 1–19.
[40] J. Xie, S. Kelley, B.K. Szymanski, Overlapping community detection in networks: the state-of-the-art and comparative study, ACM Comput. Surv. 45 (2013) 1–35.
[41] C. Yang, Z. Liu, D. Zhao, M. Sun, E.Y. Chang, Network representation learning with rich text information, in: IJCAI, 2015, pp. 2111–2117.
[42] H. Yang, X. Liu, Studies on the clustering algorithm for analyzing gene expression data with a bidirectional penalty, J. Comput. Biol. 24 (2017) 689–698.
[43] H. Yang, J. Luo, Y. Fan, L. Zhu, Using weighted k-means to identify Chinese leading venture capital firms incorporating with centrality measures, Inf. Process. Manag. 57 (2020) 102083.
[44] H. Yang, Z. Zhuang, W. Pan, A graph convolutional neural network for gene expression data analysis with multiple gene networks, Stat. Med. 40 (2021) 5547–5564.
[45] L. Yang, X. Cao, D. He, C. Wang, X. Wang, W. Zhang, Modularity based community detection with deep learning, in: IJCAI, 2016, pp. 2252–2258.
[46] Z. Yang, J. Guo, K. Cai, J. Tang, J.Z. Li, L. Zhang, Z. Su, Understanding retweeting behaviors in social networks, in: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, 2010, pp. 1633–1636.
[47] B. Yu, Z. Zheng, J. Dai, K-dghc: a hierarchical clustering method based on k-dominance granularity, Inf. Sci. 632 (2023) 232–251.
[48] M. Yuan, Y. Lin, Model selection and estimation in regression with grouped variables, J. R. Stat. Soc., Ser. B, Stat. Methodol. 68 (2006) 49–67.
[49] P. Zhang, J. Chen, C. Che, L. Zhang, B. Jin, Y. Zhu, Iea-gnn: anchor-aware graph neural network fused with information entropy for node classification and link prediction, Inf. Sci. 634 (2023) 665–676.
[50] W. Zhu, C. Chen, B. Peng, Unified robust network embedding framework for community detection via extreme adversarial attacks, Inf. Sci. 643 (2023) 119200.