Exploring Temporal Community Structure via Network Embedding

Tianpeng Li, Wenjun Wang, Pengfei Jiao, *Member, IEEE*, Yinghui Wang, Ruomeng Ding, Huaming Wu, *Senior Member, IEEE*, Lin Pan, and Di Jin

***Abstract*—Temporal community detection is helpful to discover and analyze significant groups or clusters hidden in dynamic networks in the real world. A variety of methods, such as modularity optimization, spectral method, and statistical network model, has been developed from diversified perspectives. Recently, network embedding-based technologies have made significant progress, and one can exploit deep learning superiority to network tasks. Although some methods for static networks have shown promising results in boosting community detection by integrating community embedding, they are not suitable for temporal networks and unable to capture their dynamics. Furthermore, the dynamic embedding methods only model network varying without considering community structures. Hence, in this article, we propose a novel unsupervised dynamic community detection model, which is based on network embedding and can effectively discover temporal communities and model dynamic networks. More specifically, we propose the community prior by introducing the Gaussian mixture model (GMM) in the variational autoencoder, which can obtain community information and better model the evolutionary characteristics of community structure and node embedding by utilizing the variant of gated recurrent unit (GRU). Extensive experiments conducted in real-world and artificial networks demonstrate that our proposed model has a better effect on improving the accuracy of dynamic community detection.**

***Index Terms*—Community detection, dynamic networks, network embedding, temporal community structure, variational autoencoder (VAE).**

Manuscript received 6 January 2022; accepted 11 April 2022. Date of publication 4 May 2022; date of current version 17 October 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 61902278, Grant 61806061, Grant 62003120, and Grant 62071327; in part by the Intelligent Manufacturing Special Fund Project of Tianjin, China, under Grant 20201198; in part by the National Key Research and Development Program of Jiangxi, China, under Grant 20212ABCO3W12; in part by the Shenzhen Sustainable Development Project under Grant KCXFZ20201221173013036; and in part by the National Key Research and Development Program of China under Grant 2020YFC1522600. This article was recommended by Associate Editor Q. Liu. *(Corresponding author: Wenjun Wang.)*

Tianpeng Li, Wenjun Wang, Yinghui Wang, Ruomeng Ding, and Di Jin are with the College of Intelligence and Computing, Tianjin University, Tianjin 300350, China (e-mail: ltpnimeia@tju.edu.cn; wjwang@tju.edu.cn; wangyinghui@tju.edu.cn; dingruomeng@tju.edu.cn; jindi@tju.edu.cn).

Pengfei Jiao is with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China (e-mail: pjiao@hdu.edu.cn).

Huaming Wu is with the Center for Applied Mathematics, Tianjin University, Tianjin 300072, China (e-mail: whming@tju.edu.cn).

Lin Pan is with the College of Intelligence and Computing, Tianjin University, Tianjin 300350, China, and also with the Marine Science and Technology, Tianjin University, Tianjin 300072, China (e-mail: linpan@tju.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2022.3168343.

Digital Object Identifier 10.1109/TCYB.2022.3168343

I. Introduction

DYNAMIC networks have attracted attention for a long time due to their ability to represent many real-world systems and their diversity, including social networks of individual connections [1], citation networks among academic papers [2], [3], and biological networks such as protein–protein interaction networks [4], [5]. Dynamic or temporal community detection aims to discover meaningful groups or clusters hidden in dynamic networks [6], [7], which is beneficial to a variety of applications, such as information spreading [8] and link prediction [9]–[11]. As an integral part of temporal community detection, community evolution focuses on analyzing changing behaviors and patterns. It can also reveal the evolutionary characteristics of the network, which quantifies the transitional relationships of communities between consecutive snapshots [12]. Therefore, it can model the real complex system better and has more challenges compared to community detection in static networks [7].

Many works have been proposed to detect the community structure and analyze its evolution for the dynamic network, including modularity optimization [13], spectral clustering [14], multiobjective evolutionary algorithm [15], [16], nonnegative matrix factorization [17], and generation models [18]. Mucha *et al.* [13] denoted a new modularity metric for time-dependent and multiplex networks. Liu *et al.* [14] proposed the PisCES with eigenvector smoothing based on the spectral method. Folino and Pizzuti [19] first extended the multiobjective approach for the dynamic networks and Zeng *et al.* [20] proposed the consensus community and a particle swarm optimization algorithm further. Ma and Dong [17] constructed two evolutionary nonnegative matrix factorization frameworks for community detection in dynamic networks. Dynamic stochastic block and latent space mode are the two types of generative methods for this problem. All these methods of temporal community detection are designed specially and have their advantages, respectively.

Recently, as a key part of deep learning [21], network embedding has achieved widespread success in complex network analysis [22], [23], which embeds each node of the network into a low-dimensional space and applies it to downstream tasks, such as the classification and link prediction. Considerable effort has been committed to developing network embedding techniques, including matrix factorization, random walk, and graph neural networks [24]. However, these methods are designed for static networks and fail to analyze the dynamic networks. To extract rich temporal information of dynamic networks, some novel methods are put forward to mine the underlying network dynamics of long-term evolution [25]–[28]. Meanwhile, some community-based embedding methods in static networks have manifested that network embedding cannot only integrate node and label information to improve the ability of latent representation but also promote the performance of community detection [29]–[32]. However, they are not suitable for the temporal networks for they lose the modeling of dynamic behaviors.

To deal with the dynamic networks, some important representation methods for dynamic network embedding are developed [33], [34]. For example, VGRNN [35] integrates high-level latent random variables into a graph recurrent neural network, which can learn interpretable latent representations and generate dynamic networks, and the learned embedding can be used to predict future links. A comprehensive review about the network embedding in dynamic networks can be referred in [36]. Usually, with the embedding result of the dynamic network, it can further calculate the community structure with the node embeddings via the unsupervised clustering algorithm.

This kind of two-stage mechanism may obtain wrong community results because it does not take community information into account. So it is interesting to construct a dynamic network representation method that could integrate the community structure and topological varying. Therefore, it is an urgent task to detect the temporal communities in dynamic networks via network embedding. However, it faces many challenges, for example, irregular varying and dynamic dependencies of links, the generation of embedded varying links, and the impact of temporal community structure on node embedding, which make this problem more complicated and meaningful.

To address the above-mentioned challenges, we try to model dynamic networks with community structure based on network embedding and improve temporal community detection. First, the above problem can be solved from the perspective of probability instead of using the costly two-step approach and ignoring uncertainty in the modeling process. We assume that nodes and edges are modeled from the Gaussian mixture model (GMM), which facilitates coupling community assignment and network embedding into a joint probability distribution. Because the GMM is a multimodal distribution, which is required by the community assignment, while the GMM is more computationally efficient and more accurate than other multimodal distributions like Dirichlet distribution due to the immature reparameterize technique. By combining the variational autoencoder (VAE) [37] with a community prior, that is, the GMM distribution, we can model the uncertainty of the network and the community membership simultaneously. As for modeling the dynamic characteristics of the network, inspired by VGRNN, we can utilize the RNN [38] structure to capture the dynamic information. More specifically, we construct a gated recurrent unit (GRU) instead of RNN on the varied relationship of nodes to capture the complex dependencies in the embedding space. By the combination of GRU and VAE, we propose a deep Bayesian model called the variational graph recurrent gmm autoencoder model (VGRGMM), to learn the dynamic network embedding and the community membership of node simultaneously.

To sum up, this article proposes a dynamic community detection model called VGRGMM based on a variational graph autoencoder with GRU and GMM. It is an end-to-end framework for community detection in dynamic networks designed specifically to meet the above challenges. The main contributions of this article can be summarized as follows.

1) We propose a novel unsupervised dynamic community detection model VGRGMM based on graph deep learning, which can effectively model temporal community structure and network embedding.
2) The proposed model incorporates communities across the snapshots of the dynamic network into a variational autoencoder with GMM and integrates the GRU to capture temporal dynamics in the embedding space.
3) Experimental results demonstrate that VGRGMM can significantly improve the performance of temporal community detection when compared with baselines on both synthetic datasets and real-world networks.

The remainder of this article proceeds as follows. Section II presents the related work on dynamic community detection and dynamic network embedding. Section III formally describes our proposed method and the optimization algorithm. Then, we validate our approach by analyzing extensive experiments on both synthetic and real-world datasets in Section IV, a case study is also given and Section V concludes this article. Section VI shows some limitations and points out future directions.

II. Related Work

A. Dynamic Community Detection

There is a taxonomy of dynamic community detection methods, including incremental-based clustering, evolutionary-based clustering, and generative-based clustering [19], [39], [40]. Incremental-based clustering converts community evolution into changes in nodes and edges between snapshots, thereby incrementally updating the attributes of nodes by defining different objective functions. However, this type of method suffers from error accumulation issues due to the excessive reliance on the defined objective functions, which are sensitive to the noise accordingly.

The main idea of evolutionary-based clustering is to incorporate the community structure information obtained at the current time step with that obtained at one or more previous time steps [13], [19]. A representative method is DYNMOGA [19], which formalizes community detection of dynamic networks as a multiobjective optimization problem. Xu *et al.* [39] presented AFFECT, an evolutionary clustering framework that uses shrinkage estimation to adaptively estimate the optimal balance parameters. Liu *et al.* [40] proposed a multiobjective evolutionary clustering algorithm DECS, in which an evolving community migration operator is developed to ensure that neighbor nodes are grouped. However, the high computational complexity caused by repeated calculations is an unavoidable issue in these methods.

Starting from the network generation mechanism, generative-based clustering methods consider that community transfer of nodes obeys the Hidden Markov hypothesis to transform community detection into the parameter estimation problem of the probabilistic model in dynamic networks. Furthermore, there is a dichotomy of the methods: latent space model [41] and dynamic stochastic block model (DSBM) [18], which are of great significance in theoretical interpretation and generative capability. Tang and Yang [12] modeled the transition of community memberships for individual nodes and employed a Bayesian treatment for parameter estimation for finding communities and their evolution in a dynamic network. Nevertheless, these models still have high complexity and intractable optimization problems due to a huge number of parameters in a probabilistic model, thus developing an excellent algorithm that is most suitable for dynamic community detection remains largely unexplored.

B. Dynamic Network Embedding

Due to the powerful representation capabilities of neural networks, dynamic network embedding based on deep neural networks has achieved impressive results. First, inspired by static graph embedding methods [42], some methods have been proposed via adding temporal regularization into the objective function. Wang *et al.* extended SDNE [43], a classical static network embedding model, into a dynamic embedding model called DynGEM [44]. It learns the node embeddings by initializing the current embeddings with the previous embeddings. However, it only considers the information of the previous snapshots, while ignoring historical information.

In order to better capture the historical information, Lipton *et al.* [38] utilized the temporal information in each snapshot $t_1, t_2, \ldots, t_{l-1}$ when calculating the embedding at snapshot t_l based on RNN, that is, the representation at t_l is not only dependent on the previous snapshot. Goyal *et al.* [25] proposed dyngraph2vec, which consists of DynAE, DynRNN, and DynAERNN, to capture the temporal patterns of the network based on the variational autoencoder framework. DynAE uses multiple fully connected layers for both the encoder and decoder, while DynRNN adopted LSTM as both the encoder and decoder and DynAERNN utilized a fully connected encoder to obtain low-dimensional hidden representations as the input of LSTM to learn the node embedding and the decoder is similar to DynAE. As the internal transition structure of RNN is completely deterministic, these methods cannot model the uncertainty of nodes. To overcome the obstacle, Zhao *et al.* [45] proposed BurstGraph that uses two RNN-based variational autoencoders to model the graph evolution mechanism as well as the uncertainty of node embedding. Hasanzadeh *et al.* [35] developed VGRNN, a variational model that incorporates additional latent random variables into the hidden states of a graph RNN to better capture the dynamic changes of the network, and it is a refined combination of GNN and RNN in the aspect of dynamic node embedding learning. But in the aspect of dynamic community detection, it inevitably needs additional clustering methods to obtain community memberships. Such works mainly focus on downstream tasks, for example, link prediction and node classification, which first obtain the low-dimensional node embedding, and then execute related algorithms on the embedding. However, for dynamic community detection, it is inappropriate to execute clustering algorithms on embedding results in the same way, because the embedding may lose too much community information of the network.

C. Community Detection Based on Network Embedding

Several embedding methods that combine node embedding and community embedding at the same time have been proposed for community detection. Cavallari *et al.* [29] proposed a framework that forms a closed loop among node embedding, community embedding, and community detection, where node embedding can be conducive to outputting meaningful community embedding and promoting community detection, while community embedding can optimize node embedding conversely. A community-based variational autoencoder model called ComVAE was proposed in [31] to learn node embedding, where community information obtained from community detection algorithms, namely, LPA and Informap, was utilized to enhance node embeddings. Choong *et al.* [30] presented VGAECD from the perspective of variational autoencoders. This method replaces the single prior of variational autoencoders with GMM, which is suitable for community detection in essence. Unfortunately, they are unable to model temporal characteristics, which plays a key role in modeling dynamic networks.

Motivated by the above related works, we propose the VGRGMM model, which learns the dynamic network embedding and community membership of node simultaneously by combining the Bayesian model and the deep structure into it. In Section III, we will introduce the structure of VGRGMM, the inference of it, and the optimization algorithm.

III. Proposed Method

In this section, we introduce the notations used in this article, the framework and its detailed process of the VGRGMM model, and the optimization algorithm.

A. Notations and Problem

In this article, we focus on the undirected and unweighted dynamic network and denote it as $\mathcal{G} = \{G^{(1)}, G^{(2)}, \ldots, G^{(T)}\}$, where T represents the number of snapshots in the dynamic network, and $G^{(t)} = (V^{(t)}, E^{(t)})$ is the snapshot graph at time step t composed of a set of nodes $V^{(t)}$ and a set of edges $E^{(t)}$. Node number is $N^{(t)} = |V^{(t)}|$ and edge number is $M^{(t)} = |E^{(t)}|$. The connections in $G^{(t)}$ can be represented by the symmetric adjacency matrix $\mathbf{A}^{(t)} \in \{0, 1\}^{N^{(t)} \times N^{(t)}}$, where $\mathbf{A}^{(t)}_{i,j} = 1$ means nodes $v_i \in V^{(t)}$ and $v_j \in V^{(t)}$ have a edge, and $\mathbf{A}^{(t)}_{i,j} = 0$ otherwise. If node features exist in $G^{(t)}$, represent them as a matrix $\mathbf{X}^{(t)}$, whose ith row is the feature vector of node $v_i \in V^{(t)}$ at time step t; otherwise, we set $\mathbf{X}^{(t)} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Network embedding

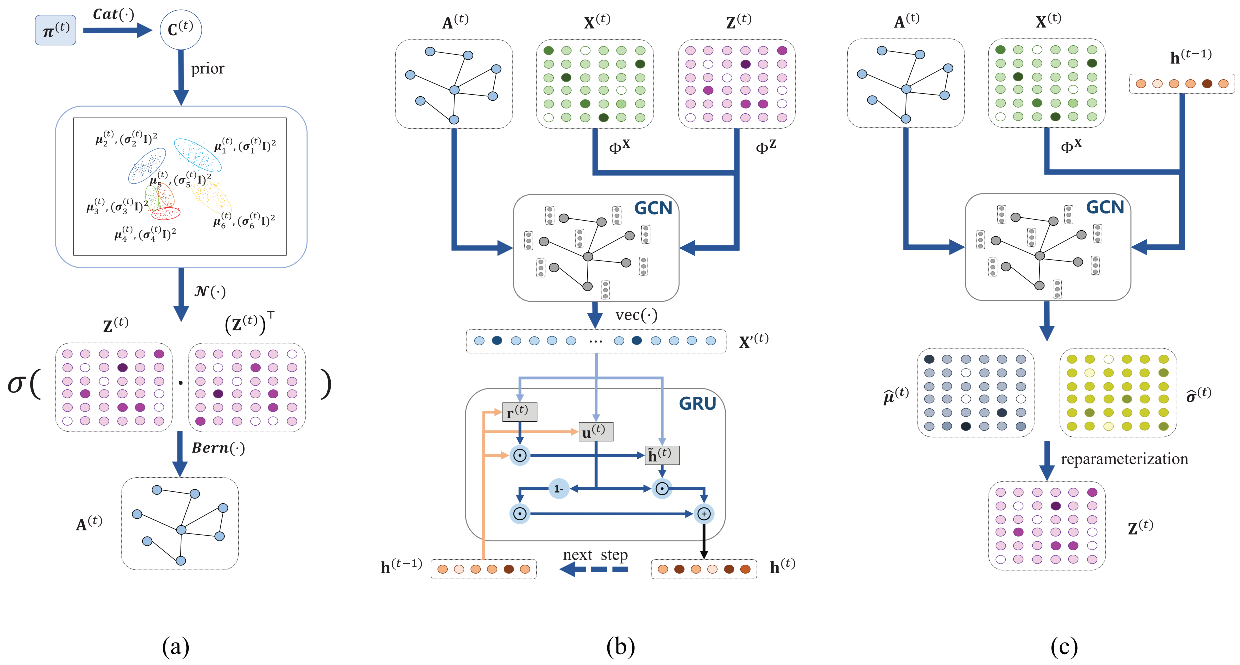


Fig. 1. Framework of VGRGMM on snapshot t. It takes the topological structure $A^{(t)}$ and attributes $X^{(t)}$ (if any) as input, the GCN as encoder, a GMM to detect the community structure, a GRU to reveal the temporal behaviors, and a decoder to reconstruct the network structure. (a) Generation. (b) Evolution. (c) Inference.

on dynamic networks aims to generate a set of node representation vectors whose dimension D is much smaller than $N^{(t)}$ for each snapshot $G^{(t)}$. We use the matrix $\mathbf{Z}^{(t)} \in \boldsymbol{R}^{N^{(t)} \times D}$ to represent the stacked embedding vectors of $G^{(t)}$. Community detection in a dynamic network is to partition the nodes in every $G^{(t)}$ of $\mathcal{G}$ into $K^{(t)}$ groups $\mathcal{C}^{(t)} = \{C_1^{(t)}, C_2^{(t)}, \ldots, C_{K^{(t)}}^{(t)}\}$ so that $\cup_{k=1}^{K^{(t)}} C_k^{(t)} = V^{(t)}$. Community detection via network embedding groups nodes close to each other within the same cluster while nodes in different clusters are far away in terms of network structure. In this article, we do not consider overlapping communities [46], that is, $C_k^{(t)} \cap C_l^{(t)} = \emptyset \quad \forall k \neq l$. We summarize main notations and their descriptions as listed in Table I.

TABLE I
NOTATIONS AND THEIR DESCRIPTIONS

Notation	Description
$\mathcal{G}$	the undirected, unweighted dynamic network
$G^{(t)}$	the snapshot network at time step t
$V^{(t)}, E^{(t)}$	the node and edge sets of $G^{(t)}$
$N^{(t)}$,$M^{(t)}$	the node and edge numbers of $G^{(t)}$
T	the number of snapshots in $\mathcal{G}$
$\mathbf{A}^{(t)}$	the adjacency matrix of $G^{(t)}$
$\mathbf{X}^{(t)}$	the feature matrix of $G^{(t)}$
$\mathbf{Z}^{(t)}$	the embedding matrix of $G^{(t)}$
D	the embedding dimension
$\mathcal{C}^{(t)}$	the partition of $V^{(t)}$
$C_k^{(t)}$	the k-th community of $c_i^{(t)}$
$K^{(t)}$	the number of communities in $\mathcal{C}^{(t)}$
$c_i^{(t)}$	the community v_i belongs to at time step t
$\boldsymbol{\pi}^{(t)}$	the parameter of the generative categorical distribution
$\boldsymbol{\mu}_{c_i^{(t)}}^{(t)}, \boldsymbol{\sigma}_{c_i^{(t)}}^{(t)}$	the parameters of the generative Gaussian distribution
$b_{i,j}^{(t)}$	the parameter of the generative Bernouli distribution
$\mathbf{C}^{(t)}$	the community assignment matrix of $G^{(t)}$
$\Phi^{\mathbf{X}}$, $\Phi^{\mathbf{Z}}$	the Multi-layer perceptron models
$\mathbf{h}^{(t)}$	the hidden state representation of GRU
$\hat{\boldsymbol{\mu}}_i^{(t)}$, $\hat{\boldsymbol{\sigma}}_i^{(t)}$	the parameters of the inferred Gaussian distribution
$\mathcal{S}_i^{(t)}$	the set of Monte Carlo samples for v_i at time step t
$\gamma_{c_i^{(t)}}^{(t)}$	the simple indicator for $q(\mathbf{C}_i^{(t)}\vert\mathbf{A}^{(t)}, \mathbf{X}^{(t)})$

B. VGRGMM Framework

To detect the temporal communities in the dynamic network based on the network embedding, we propose the VGRGMM model, which generates the first snapshot as a static network and models the dynamics of snapshot $t \geq 2$ across the network. The model is based on the encoder–decoder framework. In the encoder part, we take the graph convolutional networks (GCNs) [47] to project the nodes into the latent space. Considering the inherent mechanism and community structure in the network, we place the mixture Gaussian distribution on the latent variables under the VAE [37]. With this design, it can improve the ability of community detection at snapshots. Besides, the most important part is modeling the dynamic varying of the network. Here, we propose to take the GRU to analyze the nodes embedding and model the temporal dependence across the snapshots.

We show the process of snapshot t of the VGRGMM depicted in Fig. 1. It includes the GCN part with the adjacency matrix $\mathbf{A}^{(t)}$ and feature matrix $\mathbf{X}^{(t)}$. The VAE part with the GMM prior embeds the nodes into latent space, the modified GRU part on the temporal dynamics and the decoder part. Here, the modified GRU could capture the dynamic varying of networks in the embedding space with the hidden state embedding $\mathbf{h}^{(t)}$. So our model can effectively model the dynamic network and its temporal varying and capture its community structure.

1) Generation: At the snapshot t, model the probability of each node v_i belonging to a community $c_i^{(t)}$ as follows:

$$c_i^{(t)} \sim \boldsymbol{Cat}\left(\boldsymbol{\pi}^{(t)}\right) \tag{1}$$

where $\boldsymbol{\pi}^{(t)}$ is the parameter of the categorical distribution $\boldsymbol{Cat}(\boldsymbol{\pi}^{(t)})$. The kth entry of $\boldsymbol{\pi}^{(t)} \in \boldsymbol{R}_{+}^{K^{(t)}}$ is the prior probability for cluster $C_k^{(t)}$ and it meets $\sum_{k=1}^{K^{(t)}} \pi_k^{(t)} = 1$. Next, for node v_i, we generate a latent vector based on the community in which it originates

$$\mathbf{Z}_i^{(t)}|c_i^{(t)} \sim \mathcal{N}\left(\boldsymbol{\mu}_{c_i^{(t)}}^{(t)}, (\boldsymbol{\sigma}_{c_i^{(t)}}^{(t)}\mathbf{I})^2\right) \tag{2}$$

where $\boldsymbol{\mu}_{c_i^{(t)}}^{(t)}$ and $\boldsymbol{\sigma}_{c_i^{(t)}}^{(t)}$ are the mean and standard deviation of the Gaussian distribution corresponding to cluster $c_i^{(t)}$, respectively, and $\mathbf{I}$ is an identity matrix. Then, for each pair of nodes (v_i, v_j), their connection state at time step t is generated following a Bernoulli distribution:

$$\mathbf{A}_{i,j}^{(t)}|\mathbf{Z}^{(t)} \sim \boldsymbol{Bern}\left(b_{i,j}^{(t)}\right) \tag{3}$$

where $b_{i,j}^{(t)}$ is a function of $\mathbf{Z}^{(t)}$, and it can be any highly flexible function such as neural networks. Here, we follow VGAE [48] and adopt a inner product decoder as the implement of (3). More precisely, we have

$$p\left(\mathbf{A}_{i,j}^{(t)} = 1|\mathbf{Z}_i^{(t)}, \mathbf{Z}_j^{(t)}\right) = \sigma\left(\mathbf{Z}_i^{(t)}\left(\mathbf{Z}_j^{(t)}\right)^{\top}\right) \tag{4}$$

where $\sigma(\cdot)$ denotes a nonlinear function, such as sigmoid function. The joint probability for all node pairs at time step t is as follows:

$$p\left(\mathbf{A}^{(t)}|\mathbf{Z}^{(t)}\right) = \prod_{i=1}^{N^{(t)}}\prod_{j=1}^{N^{(t)}} p\left(\mathbf{A}_{i,j}^{(t)}|\mathbf{Z}_i^{(t)}, \mathbf{Z}_j^{(t)}\right). \tag{5}$$

Represent the community memberships via the community assignment matrix $\mathbf{C}^{(t)} \in \{0, 1\}^{N^{(t)} \times K^{(t)}}$ of which ith row is a one-hot vector and $\mathbf{C}_{i,c_i^{(t)}}^{(t)} = 1$. According to the entire generative process above, $\mathbf{A}^{(t)}$ and $\mathbf{C}^{(t)}$ are independently conditioned on $\mathbf{Z}^{(t)}$. Thus, the joint probability of them at all time steps can be factorized as

$$\begin{aligned} &p\left(\mathbf{A}^{(\le T)}, \mathbf{Z}^{(\le T)}, \mathbf{C}^{(\le T)}\right) \\ &= \prod_{t=1}^{T} p\left(\mathbf{A}^{(t)}|\mathbf{Z}^{(t)}\right)p\left(\mathbf{Z}^{(t)}|\mathbf{C}^{(t)}\right)p(\mathbf{C}^{(t)}) \\ &= \prod_{t=1}^{T}\left(\prod_{i=1}^{N^{(t)}}\prod_{j=1}^{N^{(t)}} p\left(\mathbf{A}_{i,j}^{(t)}|\mathbf{Z}_i^{(t)}, \mathbf{Z}_j^{(t)}\right)\prod_{i=1}^{N^{(t)}} p\left(\mathbf{Z}_i^{(t)}|\mathbf{C}_i^{(t)}\right)\prod_{i=1}^{N^{(t)}} p\left(\mathbf{C}_i^{(t)}\right)\right) \end{aligned} \tag{6}$$

where $\mathbf{A}^{(\le T)}$ is the simpler indicator for sequence $\{\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \ldots, \mathbf{A}^{(\le T)}\}$. The indicators $\mathbf{Z}^{(\le T)}$, $\mathbf{C}^{(\le T)}$, and later $\mathbf{X}^{(\le T)}$ are in the same manner.

2) Evolution: We follow GCRN [49] and modify GRU to model the dynamic dependency among snapshot networks. We first employ a two-layer GCN to fuse the information of network structure, features, and latent vectors

$$\mathbf{X}'^{(t)} = \text{GCN}_2\left(\mathbf{A}^{(t)}, \text{GCN}_1\left(\mathbf{A}^{(t)}, \left[\overline{\Phi^{\mathbf{X}}(\mathbf{X}^{(t)}), \Phi^{\mathbf{Z}}(\mathbf{Z}^{(t)})}\right]\right)\right) \tag{7}$$

where $[\overline{\cdot}]$ is the concatenation operator. $\Phi^{\mathbf{X}}$ and $\Phi^{\mathbf{Z}}$ are multilayer perceptron models that operate independently on each node and extract features from $\mathbf{X}^{(t)}$ and $\mathbf{Z}^{(t)}$ separately. The dependency information is preserved in $\mathbf{h}^{(t)}$ learned by the GRU model

$$\mathbf{r}^{(t)} = \sigma\left(\mathbf{W}_{\mathbf{r}^{(t)}}\left[\overline{\text{vec}(\mathbf{X}'^{(t)}), \mathbf{h}^{(t-1)}}\right]\right) \tag{8}$$

$$\mathbf{u}^{(t)} = \sigma\left(\mathbf{W}_{\mathbf{u}^{(t)}}\left[\overline{\text{vec}(\mathbf{X}'^{(t)}), \mathbf{h}^{(t-1)}}\right]\right) \tag{9}$$

$$\widetilde{\mathbf{h}}^{(t)} = \tanh\left(\mathbf{W}_{\widetilde{\mathbf{h}}^{(t)}}\left[\overline{\mathbf{r}^{(t)} \circ \mathbf{h}^{(t-1)}, \text{vec}\left(\mathbf{X}'^{(t)}\right)}\right]\right) \tag{10}$$

$$\mathbf{h}^{(t)} = \left(\mathbf{1} - \mathbf{u}^{(t)}\right) \circ \mathbf{h}^{(t-1)} + \mathbf{u}^{(t)} \circ \widetilde{\mathbf{h}}^{(t)} \tag{11}$$

where the vectorization operator $\text{vec}(\cdot)$ splices all the rows of the input matrix to a single vector and $\circ$ is the Hadamard product operator. $\mathbf{W}_{\mathbf{r}^{(t)}}$, $\mathbf{W}_{\mathbf{u}^{(t)}}$, and $\mathbf{W}_{\widetilde{\mathbf{h}}^{(t)}}$ are the parameter matrices of GRU gates. We initialize $\mathbf{h}^{(0)} = \mathbf{0}$.

3) Inference: We use $q(\cdot)$ as the variational posterior to approximate the true posterior $p(\cdot)$ in VGRGMM. According to the mean field distribution assumption, the approximate posterior $q(\cdot)$ is not only a function of $\mathbf{A}^{(t)}$ and $\mathbf{Z}^{(t)}$ but also a function of $\mathbf{h}^{(t-1)}$. $q(\cdot)$ can be factorized as follows:

$$\begin{aligned} &q\left(\mathbf{Z}^{(t)}, \mathbf{C}^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}^{(t-1)}\right) \\ &= q\left(\mathbf{Z}^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}^{(t-1)}\right)q\left(\mathbf{C}^{(t)}|\mathbf{A}^{(t)}, X^{(t)}\right). \end{aligned} \tag{12}$$

The inference model $q(\mathbf{Z}^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}^{(t-1)})$ is parameterized by a two-layer GCN of which the first layer is shared

$$q\left(\mathbf{Z}_i^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}^{(t-1)}\right) = \mathcal{N}\left(\hat{\boldsymbol{\mu}}_i^{(t)}, \left(\hat{\boldsymbol{\sigma}}_i^{(t)}\mathbf{I}\right)^2\right), \tag{13}$$

$$\hat{\boldsymbol{\mu}}^{(t)} = \text{GCN}_{\hat{\mu}}\left(\mathbf{A}^{(t)}, \text{GCN}_s\left(\mathbf{A}^{(t)}, \left[\overline{\Phi^{\mathbf{X}}\left(\mathbf{X}^{(t)}\right), \mathbf{h}^{(t-1)}}\right]\right)\right) \tag{14}$$

$$\log\hat{\boldsymbol{\sigma}}^{(t)} = \text{GCN}_{\hat{\sigma}}\left(\mathbf{A}^{(t)}, \text{GCN}_s\left(\mathbf{A}^{(t)}, \left[\overline{\Phi^{\mathbf{X}}\left(\mathbf{X}^{(t)}\right), \mathbf{h}^{(t-1)}}\right]\right)\right) \tag{15}$$

where $\hat{\boldsymbol{\mu}}^{(t)} \in \boldsymbol{R}^{N^{(t)} \times D}$ and $\hat{\boldsymbol{\sigma}}^{(t)} \in \boldsymbol{R}^{N^{(t)} \times D}$ denote the parameter matrices whose ith rows are the parameters $\hat{\boldsymbol{\mu}}_i^{(t)}$ and $\hat{\boldsymbol{\sigma}}_i^{(t)}$ of the approximate posterior $\mathcal{N}(\hat{\boldsymbol{\mu}}_i^{(t)}, (\hat{\boldsymbol{\sigma}}_i^{(t)}\mathbf{I})^2)$. While being concatenated, $\mathbf{h}^{(t-1)}$ is transformed to a matrix having the same row number $N^{(t)}$ as $\Phi^{\mathbf{X}}(\mathbf{X}^{(t)})$ by being duplicated $N^{(t)}$ times. Here, the reparameterization trick is applied

$$\mathbf{Z}_i^{(t)} = \hat{\boldsymbol{\mu}}_i^{(t)} + \hat{\boldsymbol{\sigma}}_i^{(t)} \circ \boldsymbol{\epsilon}^{(t)} \tag{16}$$

where $\boldsymbol{\epsilon}^{(t)}$ is an auxiliary noise variable $\epsilon^{(t)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

The computation of $q(\mathbf{C}^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)})$ follows the equation from [50]:

$$\begin{aligned} q\left(\mathbf{C}_i^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}\right) &= p\left(c_i^{(t)}|\mathbf{Z}_i^{(t)}\right) \\ &= \frac{p\left(c_i^{(t)}\right)p\left(\mathbf{Z}_i^{(t)}|c_i^{(t)}\right)}{\sum_{c_i'^{(t)}=1}^{K^{(t)}} p\left(c_i'^{(t)}\right)p\left(\mathbf{Z}_i^{(t)}|c_i'^{(t)}\right)}. \end{aligned} \tag{17}$$

Since $q(\mathbf{Z}^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}^{(t-1)})$ is closely related to $\boldsymbol{h}^{(t-1)}$ that is a function of $\mathbf{A}^{(t-1)}$, $\mathbf{X}^{(t-1)}$, and $\mathbf{Z}^{(t-1)}$ as shown in (12),

$q(\mathbf{Z}^{(\leq T)}, \mathbf{C}^{(\leq T)}|\mathbf{A}^{(\leq T)}, \mathbf{X}^{(\leq T)})$ can be further decomposed as follows:

$$\begin{aligned} q&\left(\mathbf{Z}^{(\leq T)}, \mathbf{C}^{(\leq T)}|\mathbf{A}^{(\leq T)}, \mathbf{X}^{(\leq T)}\right) \\ &= \prod_{t=1}^{T} q\left(\mathbf{Z}^{(t)}, \mathbf{C}^{(t)}|\mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}, \mathbf{Z}^{(<t)}\right) \\ &= \prod_{t=1}^{T} q\left(\mathbf{Z}^{(t)}, \mathbf{C}^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}^{(t-1)}\right). \end{aligned} \tag{18}$$

In the following description, for simplicity, we use $\Delta^{(\leq T)}$ and $\Delta^{(t)}$to represent $\mathbf{Z}^{(\leq T)}, \mathbf{C}^{(\leq T)}|\mathbf{A}^{(\leq T)}, \mathbf{X}^{(\leq T)}$ and $\mathbf{Z}^{(t)}, \mathbf{C}^{(t)}|\mathbf{A}^{(\leq t)}, \mathbf{X}^{(\leq t)}, \mathbf{Z}^{(<t)}$, respectively.

C. Optimization

To optimize $q(\Delta^{(\leq T)})$ on approximating the true posterior $p(\Delta^{(\leq T)})$, the KL divergence $\mathbb{D}_{\mathrm{KL}}(q(\Delta^{(\leq T)})||p(\Delta^{(\leq T)}))$ needs to be minimized, which is equivalent to the maximization of the difference between the log likelihood $p(\mathbf{A}^{(\leq T)}, \mathbf{Z}^{(\leq T)}, \mathbf{C}^{(\leq T)})$ and KL divergence called evidence lower bound (ELBO)

$$\begin{aligned} \mathcal{L}_{\mathrm{ELBO}} &= \log p\left(\mathbf{A}^{(\leq T)}|\mathbf{Z}^{(\leq T)}, \mathbf{C}^{(\leq T)}\right) \\ &\quad - \mathbb{D}_{\mathrm{KL}}\left(q\left(\Delta^{(\leq T)}\right)||p\left(\Delta^{(\leq T)}\right)\right) \\ &= \mathbb{E}_{q(\Delta^{(\leq T)})}\left[\log p\left(\mathbf{A}^{(\leq T)}|\mathbf{Z}^{(\leq T)}, \mathbf{C}^{(\leq T)}\right)\right. \\ &\qquad \left. - \log q\left(\Delta^{(\leq T)}\right) + \log p\left(\Delta^{(\leq T)}\right)\right] \\ &= \mathbb{E}_{q(\Delta^{(\leq T)})}\left[\log p\left(\mathbf{A}^{(\leq T)}, \mathbf{Z}^{(\leq T)}, \mathbf{C}^{(\leq T)}\right)\right. \\ &\qquad \left. - \log q\left(\Delta^{(\leq T)}\right)\right]. \end{aligned} \tag{19}$$

Therefore, we learn the parameters of the generative and inference models tied jointly by maximizing the variational lower bound L_{ELBO}, which combines community and temporal information together. The objective function of L_{ELBO} then can be written based on (6) and (12) as

$$\begin{aligned} \mathcal{L}_{\mathrm{ELBO}} &= \sum_{t=1}^{T} \mathbb{E}_{q(\Delta^{(t)})}\left[\log \frac{p(\mathbf{A}^{(t)}|\mathbf{Z}^{(t)})p(\mathbf{Z}^{(t)}|\mathbf{C}^{(t)})p(\mathbf{C}^{(t)})}{q(\Delta^{(t)})}\right] \\ &= \sum_{t=1}^{T} \mathbb{E}_{q(\Delta^{(t)})}\left[\log p\left(\mathbf{A}^{(t)}|\mathbf{Z}^{(t)}\right)p\left(\mathbf{Z}^{(t)}|\mathbf{C}^{(t)}\right)p\left(\mathbf{C}^{(t)}\right)\right. \\ &\qquad - \log q\left(\mathbf{Z}^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}^{(t-1)}\right)q \\ &\qquad \left. \times\left(\mathbf{C}^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}\right)\right] \\ &= \sum_{t=1}^{T} \mathbb{E}_{q(\Delta^{(t)})}\left[\sum_{i=1}^{N^{(t)}}\left[\sum_{j=1}^{N^{(t)}} \log p\left(\mathbf{A}_{i,j}^{(t)}|\mathbf{Z}_i^{(t)}, \mathbf{Z}_j^{(t)}\right)\right.\right. \\ &\qquad + \log p\left(\mathbf{Z}_i^{(t)}|\mathbf{C}_i^{(t)}\right) + \log p\left(\mathbf{C}_i^{(t)}\right) \\ &\qquad - \log q\left(\mathbf{Z}_i^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}, \mathbf{h}^{(t-1)}\right) \\ &\qquad \left.\left. - \log q\left(\mathbf{C}^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)}\right)\right]\right]. \end{aligned} \tag{20}$$

Algorithm 1 Learning Algorithm of VGRGMM

Input: iteration number L, adjacency matrices $\mathbf{A}^{(\leq T)}$ and feature matrices $\mathbf{X}^{(\leq T)}$.

Output: embedding matrices $\mathbf{Z}^{(\leq T)}$ and community assignment matrices $\mathbf{C}^{(\leq T)}$.

```
 1: Initialization: $\boldsymbol{\pi}^{(t)} \sim \boldsymbol{\mathcal{U}}(\mathbf{0}, \mathbf{1}), \forall 1 \leq t \leq T; \mathbf{h}^{(0)} = \mathbf{0}$;
 2: for $t = 1, ..., T$ do
 3:     Get Gaussian parameters $\hat{\boldsymbol{\mu}}^{(t)}$ and $\hat{\boldsymbol{\sigma}}^{(t)}$;    ▷ Eq. (13)
 4:     Generate embedding matrix $\mathbf{Z}_i^{(t)}$;    ▷ Eq. (16)
 5:     Learn GRU hidden state $\mathbf{h}^{(t)}$;    ▷ Eq. (9)
 6:     for each $v_i \in V^{(t)}$ do
 7:         Sample a community $\mathbf{C}_i^{(t)} \sim \boldsymbol{Cat}(\boldsymbol{\pi}^{(t)})$;
 8:         Attain a MC sample set $\mathcal{S}_i^{(t)} \subseteq V^{(t)}$ [37];
 9:         for each $v_s \in \mathcal{S}_i^{(t)}$ do
10:             Reconstruct connection $\mathbf{A}_{i,s}^{(t)}$;    ▷ Eq. (3)
11:         end for
12:     end for
13: end for
14: Compute $\mathcal{L}_{ELBO}$;    ▷ Eq. (21)
15: Maximize $\mathcal{L}_{ELBO}$ via stochastic gradient descent and
    update parameters of VGRGMM.
16: Redo Line 2-15 until the iteration limit $L$ is reached.
17: Obtain community assignment $\mathbf{C}^{(\leq T)}$.    ▷ Eq. (17)
18: return $\mathbf{Z}^{(\leq T)}$ and $\mathbf{C}^{(\leq T)}$
```

According to (1)–(3) and (13), $\mathcal{L}_{\mathrm{ELBO}}$ can be further derived into the following equation concretely by using SGVB [37] estimator:

$$\begin{aligned} &\mathcal{L}_{\mathrm{ELBO}} \\ &= \sum_{t=1}^{T}\sum_{i=1}^{N^{(t)}}\left[\frac{1}{|\mathcal{S}_i^{(t)}|}\sum_{v_s \in \mathcal{S}_i^{(t)}}\left[\mathbf{A}_{i,s}^{(t)} \log b_{i,s}^{(t)}\right.\right. \\ &\qquad \left. + \left(1 - \mathbf{A}_{i,s}^{(t)}\right)\log\left(1 - \log b_{i,s}^{(t)}\right)\right] \\ &+ \sum_{c_i^{(t)}=1}^{K^{(t)}}\sum_{d=1}^{D}\left[\gamma_{c_i^{(t)}}^{(t)} \log \frac{\boldsymbol{\pi}_{c_i^{(t)}}^{(t)}}{\gamma_{c_i^{(t)}}^{(t)}} + \frac{1}{2}\left(1 + \log\left(\boldsymbol{\sigma}_{c_i^{(t)},d}^{(t)}\right)^2\right)\right. \\ &\qquad - \frac{1}{2}\gamma_{c_i^{(t)}}^{(t)}\left[\log\left(\boldsymbol{\sigma}_{c_i^{(t)},d}^{(t)}\right)^2 + \frac{\left(\hat{\boldsymbol{\sigma}}_{i,d}^{(t)}\right)^2}{\left(\boldsymbol{\sigma}_{c_i^{(t)},d}^{(t)}\right)^2}\right. \\ &\qquad \left.\left.\left. + \frac{\left(\hat{\boldsymbol{\mu}}_{i,d}^{(t)} - \boldsymbol{\mu}_{c_i^{(t)},d}\right)^2}{\left(\boldsymbol{\sigma}_{c_i^{(t)},d}^{(t)}\right)^2}\right]\right]\right] \end{aligned} \tag{21}$$

where $\mathcal{S}_i^{(t)}$ is the set of Monte Carlo (MC) samples for node $v_i \in V^{(t)}$ at time step t in the SGVB estimator, $\pi_{c_i^{(t)}}^{(t)}$ denotes the prior probability of cluster $c_i^{(t)}$, and $\gamma_{c_i^{(t)}}^{(t)}$ denotes $q(\mathbf{C}_i^{(t)}|\mathbf{A}^{(t)}, \mathbf{X}^{(t)})$.

We summarize the algorithmic process of our method in Algorithm 1. Lines 3 and 4 correspond to the encoding

TABLE II
STATISTICS OF DYNAMIC NETWORKS

Dataset	T	$\overline{N}$	$\overline{M}$	$\overline{K}$	$\overline{d_{\text{avg}}}$	$\overline{\text{cc}_{\text{avg}}}$
Var	10	393	3,055	8	15.492	0.162
Synth	10	128	2,048	8	20.000	0.253
HS11	7	126	424	5	6.734	0.261
HS12	8	180	538	7	5.978	0.260
Primary	6	242	2,977	13	24.610	0.527
Workplace	8	92	175	6	3.821	0.215
Cellphone	10	400	512	-	2.562	0.014
Enron	12	151	292	-	4.014	0.408
Voles	7	1,686	4,739	-	0.804	0.077
fbmessages	6	1,899	15,648	-	2.914	0.155
iadublin	8	10,972	415,913	-	75.000	1.061
soc-wiki	12	7,118	100,811	-	28.326	0.141

(inference) procedure while lines 6–12 correspond to the decoding (generation) procedure. Line 5 is the procedure to capture the evolution dependency. Lines 14 and 15 describe the optimization procedure at each training iteration. The biggest difference between our method to other embedding-based dynamic community detection methods is that once our VGRGMM is finally trained, it can naturally assign nodes into communities for all the snapshots as line 17 indicates, where we choose the community with the largest probability.

IV. EXPERIMENTS

In this section, we introduce a variety of network datasets and baselines and four evaluation metrics on community detection. Then, we conduct several experiments to demonstrate the performance of our proposed model for the dynamic community detection task.

A. Datasets

We chose three types of datasets including 12 dynamic networks in our experiments.

1) *Synthetic Dataset:* The first generated data are according to [51] and [52], which is a dynamic network with the varying number of communities. At the first snapshot, it generates the network with community structure based on the stochastic block model, then at each snapshot $t \geq 2$, it changes the communities and generates the network structure with the parameter setting in [52]. The other synthetic dataset is from [53], which can generate the dynamic network with a power-law degree distribution, and the varying of network structure is driven by the transfer of nodes in different communities. We call these two dynamic networks Var and Synth, the statistics of which are seen in Table II, and the other parameter settings can be found in [51] and [52].
2) *Real-World Social Network With Ground Truth:* It contains four dynamic social networks, namely: a) HS11 [54]; b) HS12 [54]; c) Primary [54]; and d) Workplace [55]. Their nodes represent people, and edges indicate the communications among nodes, and the communities are defined as friends, groups, or classes.
3) *Real-World Network Without Ground Truth:* It contains Cellphone [19] (a telephone network of an organization), Enron[1] (an email communication network), Voles [56] (an interactive network of animals), Fbmessages [56] (a social network derived from the University of California), and two larger-scale networks: a) iadublin[2] and b) soc-wiki [56].

The statistics of the above dynamic networks are summarized in Table II. $\overline{N}$, $\overline{M}$, $\overline{K}$, $\overline{d_{\text{avg}}}$, and $\overline{\text{cc}_{\text{avg}}}$ represent the average number of nodes, edges, groups, mean degree, and mean clustering coefficient at all snapshots, respectively.

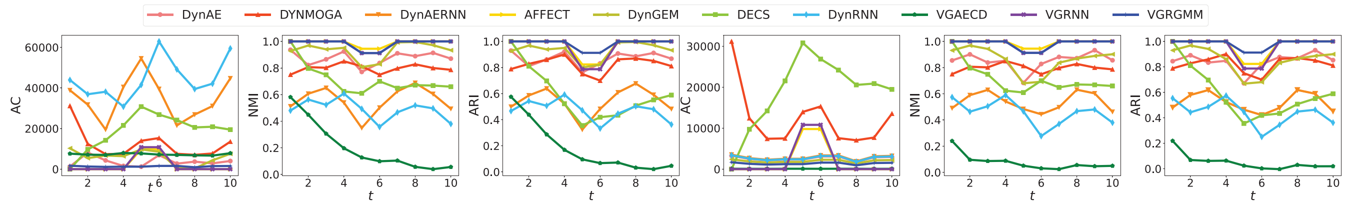
B. Baselines Setting

To make a comprehensive comparison, here we compare our VGRGMM with the popular methods of community detection in dynamic networks AFFECT [39], DYNMOGA [19], and DECS [40] (nondeep learning), and dynamic embedding methods based on deep learning, including DynGEM [44], DynAE [57], DynRNN [25], DynAERNN [25], and VGRNN [35].

1) *AFFECT [39]:* It treats evolutionary clustering as a tracking problem followed by ordinary static clustering, which involves smoothing proximities between objects over time followed by static clustering.
2) *DYNMOGA [19]:* It regards the detection of community structure in dynamic networks as a multiobjective optimization problem. The first objective searches for highly modular structures at the current time step, and the second objective tries to minimize the differences between the community structure at the current time step and that obtained at the previous snapshot.
3) *DECS [40]:* It develops a migration operator cooperating with efficient operators to ensure that nodes and their most neighbors are grouped together, and uses a genome matrix encoding the structure information of networks to expand the search space.
4) *DynGEM [44]:* It employs a deep autoencoder at its core and leverages deep learning to generate highly nonlinear embeddings and the observed network.
5) *DynAE [57]:* It addresses a community detection and network reconstruction tradeoff, by gradually and smoothly eliminating the reconstruction objective function in favor of a construction one.
6) *DynRNN&DynAERNN [25]:* DynRNN takes a set of previous graphs as input and generates the network structure as output at the next time step, thereby it can capture highly nonlinear interactions between the nodes at each time step and across multiple snapshots. Besides, it also captures temporal information through an RNN framework. The difference between DynAERNN and DynRNN is that DynAERNN captures the high number of model parameters through an autoencoder RNN.
7) *VGRNN [35]:* It develops a model that is universally compatible with potential changes in both node and edge

[1] http://www.cs.cmu.edu/ enron

[2] http://www.sociopatterns.org/datasets/



(a) (b)

Fig. 2. Experiment results on the two generated dynamic networks with the ground truth. (a) Var dataset. (b) Synth dataset. The performance results are based on the AC, NMI, and ARI, respectively. Each value is computed by the average of ten randomly generated networks with the same parameters.

sets, and each node at each snapshot is represented with a specific prior distribution.

8) *VGAECD:* It is the method of community detection with variational graph autoencoder on each snapshot network independently. It is also equivalent to the method in [30] and the special case of our VGRGMM method that ignores dependencies across the snapshots.

For nondeep learning models, the parameters in their models are set based on their original papers to guarantee the best performance. For deep learning models, we make fine-tuning the parameters in these models to obtain the best performances in different datasets. As for our model in all datasets, we set up a single recurrent hidden layer with 32 GRU units. In the encoder process, we use 2-layer GCN with a size equal to [32, 16] to devise $\hat{\boldsymbol{\mu}}^{(t)}$ and $\hat{\boldsymbol{\sigma}}^{(t)}$ and two 32-D fully connected layers for $\Phi^{\mathbf{X}}$ and $\Phi^{\mathbf{Z}}$. In addition, we train $L = 1000$ iterations with a learning rate of 0.01, or until the model converges. For that most of these methods are all set the number of communities $K^{(1)}, K^{(2)}, \dots, K^{(T)}$ of the dynamic network in advance, we set that as the true values. For the real networks without ground truth, we compute that with the popular method [58] on each snapshot.

C. Evaluation Metrics

Here, we use four widely used metrics to evaluate the performance of community detection and they are listed as follows.

1) *Normalized Mutual Information (NMI):* It measures the closeness between the computed partition of $\mathcal{C}^{(t)}$ and the true community partition $\mathcal{C}'^{(t)}$ at time step t

$$\mathrm{NMI}\left(\mathcal{C}^{(t)}, \mathcal{C}_{\mathrm{T}}^{(t)}\right) = \frac{2I\left(\mathcal{C}^{(t)}, \mathcal{C}'^{(t)}\right)}{H\left(\mathcal{C}^{(t)}\right) + H\left(\mathcal{C}'^{(t)}\right)} \tag{22}$$

where $H(\cdot)$ is the entropy function, and $I(\mathcal{C}^{(t)}, \mathcal{C}'^{(t)}) = H(\mathcal{C}^{(t)}) + H(\mathcal{C}'^{(t)}) - H(\mathcal{C}^{(t)}, \mathcal{C}'^{(t)})$.

2) *Accurate Calibration (AC) [19]:* It measures the performance with ground truth

$$\mathrm{AC} = \left\| \mathbf{C}^{(t)}\left(\mathbf{C}^{(t)}\right)^{\top} - \mathbf{C}'^{(t)}\left(\mathbf{C}'^{(t)}\right)^{\top} \right\| \tag{23}$$

where $\mathbf{C}^{(t)}$ is the learned community assignment matrix at time step t while $\mathbf{C}'^{(t)}$ is the groud truth of community assignment at the corresponding time step. The lower the AC value, the better the community detection performance.

3) *Adjusted Rand Index (ARI)*: It is a data clustering metric and is used to measure the similarity between two partitions $\mathcal{C}^{(t)}$ and the true community partition $\mathcal{C}_{\mathrm{T}}^{(t)}$

$$\mathrm{ARI} = \frac{\sum_{ij}\binom{n_{i,j}}{2} - \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{n}{2}}}{\frac{\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}}{2} - \frac{\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}}{\binom{N^{(t)}}{2}}} \tag{24}$$

where $n_{i,j}$ denotes the number of common elements between the ith subset of $\mathcal{C}^{(t)}$ and the jth subset of $\mathcal{C}'^{(t)}$, $a_i = \sum_j n_{i,j}$ and $b_j = \sum_i n_{i,j}$.

4) *Modularity (Q)*: It measures the strength of dividing the graph into communities. A graph with a high degree of modularity has dense connections between nodes within modules, but sparse connections between nodes in different modules

$$Q = \frac{1}{2M^{(t)}} \sum_{ij} \left(\mathbf{A}_{i,j} - \frac{d_i^{(t)} d_j^{(t)}}{2M^{(t)}} \right) \delta_{i,j}^{(t)} \tag{25}$$

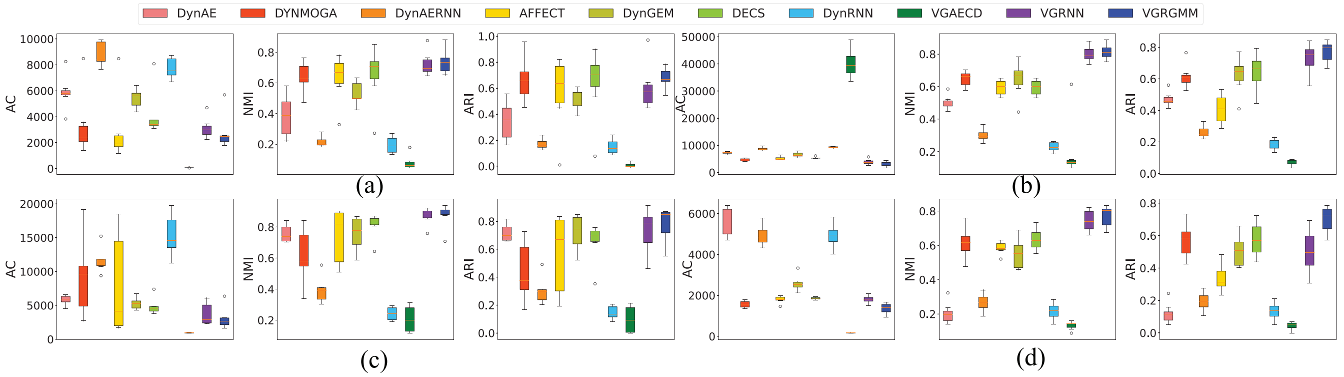
where Q is given by the sum of $\mathbf{A}_{i,j} - ([d_i^{(t)} d_j^{(t)}]/[2M^{(t)}])$ over all pairs of vertices i and j that fall in the same community. $\delta_{i,j}^{(t)} = 1$ when both nodes v_i and v_j belong to the same community at time step t, and 0 otherwise.

It should be noted that NMI, AC, and ARI are only applicable when ground truth exits, while Q can measure without ground truth. Larger values of NMI, Q, and ARI mean better performance of the method and it is the opposite for AC.

D. Dynamic Community Detection Results

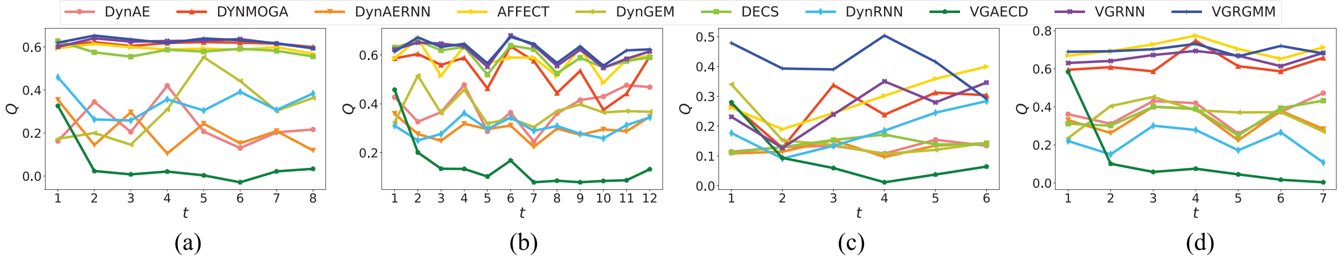
To evaluate the dynamic community detection performance of VGRGMM, we compare it with other baselines on the popular synthetic and real dynamic networks. For the dynamic networks with ground truth, we show the results based on the metrics NMI, AC, and ARI. For the dataset without labels, we evaluate different methods with the modularity Q.

1) Results on Generated Datasets: As shown in Fig. 2, it presents the community detection results of all the methods on the generated datasets Var and Synth (a large-scale network). We evaluate the methods on the NMI, AC, and ARI metrics. For the two different sizes of networks, our method VGRGMM has nearly the best performance based on all the three metrics. AFFECT and VGRNN have competitive results compared with VGRGMM and are superior to other baselines, AFFECT can automatically balance two goals on the evolutionary clustering, and VGRNN captures the temporal



(a) (b) (c) (d)

Fig. 3. Experiment results on four real-world dynamic networks with ground truth. (a) HS11. (b) HS12. (c) Primary. (d) Workplace. The box diagram shows the average value and standard deviation of the results across the snapshots of each dynamic network. Larger NMI and ARI and smaller AC mean better performance and smaller values of standard deviation represent the stable results.



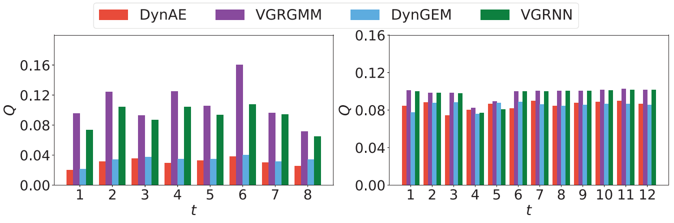
(a) (b) (c) (d)

Fig. 4. Experiment results on real-world datasets without ground truth. (a) Cellphone. (b) Enron. (c) fbmessages. (d) Voles. The results are evaluated based on Q and computed on ten independent runs.

behaviors and high dimension and nonlinear structure of the dynamic network, indicating the importance of the dynamic pattern of dynamic community detection methods. This is also been proved by the poor performance of VGAECD, which is a static method. On the other hand, the embedding methods, DynRNN and DynAERNN, only focus on modeling the dynamic varying of the structure, so they are failed to capture the community structure. It shows that the dynamic embedding methods lose the community information. DYNMOGA, the classical method of temporal community detection, has satisfactory results because it optimizes the communities and the dynamic pattern of the network with multiobjective functions. As a whole, VGRGMM can model the temporal dependence across the snapshots, capture the community structure with GMM, and reveal the nonlinear structure of the dynamic network; thus, it has better performance on the networks with node-level and community-level dynamic behaviors.

2) Results on Real-World Networks: Here, we show the results conducted on the real-world datasets, which are divided into networks with and without ground truth. Fig. 3 shows the results of different methods on four dynamic networks with the NMI, AC, and ARI metrics. To make the comparison more clear, we show the performance of all the methods on each dynamic network with a box diagram, and each box denotes the average value and standard deviation of results. VGRGMM has better performance on the three metrics compared to other baselines, and a smaller standard deviation means indicating that it has stable and smooth results across the snapshots. VGAECD has poor performance for it ignoring the temporal information. For the other baselines, none of them has competitive results with our method on all the networks, which means that VGRGMM can better handle the different types of dynamic networks in the real world due to the multimodal characteristics of GMM. We also show the results based on the modularity Q on four real dynamic networks without ground truth in Fig. 4, all methods demonstrate a stable performance across the snapshots and VGRGMM also shows the best or second performance on these networks. Furthermore, VGRGMM, AFFECT, DYNMOGA, DECS, and VGRNN all have excellent results because they either optimize the modularity as their loss function or consider the community structure in their model.

3) Results on Large-Scale Networks: In order to further verify the effect of VGRGMM on large-scale real-world networks, we compare VGRGMM and three efficient baselines: 1) DynAE; 2) DynGEM; and 3) VGRNN, on iadublin and soc-wiki datasets. All these methods can deal the large-scale networks compared with the traditional methods for temporal community detection. For that there is no community ground truth of these two dynamic networks, we also evaluate the community detection performance of our method and the baselines with Q. We also take the method [58] to compute the number of communities at each snapshot and obtain the community structure for all these methods. Fig. 5 shows the results of the modularity Q of these methods. For the large-scale and sparse dynamic networks, the Q values of all methods are very small. Generally, our VGRGMM still has some advantages over DynAE, DynGEM, and VGRNN on both networks at all the snapshots. This indicates that the deep Bayesian method VGRGMM can not only handle the



(a) (b)

Fig. 5. Results of temporal community detection of VGRGMM, DynAE, DynGEM, and VGRGMM on two large-scale networks. (a) iadublin. (b) soc-wiki.

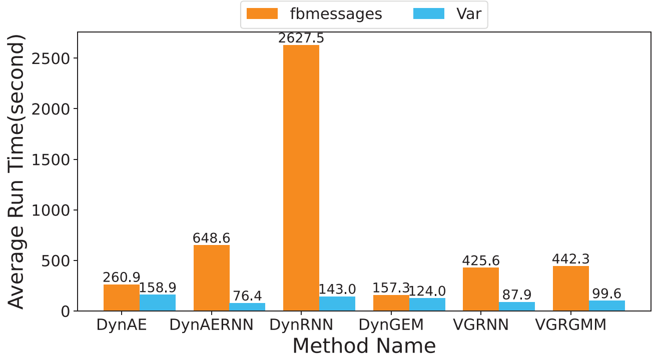
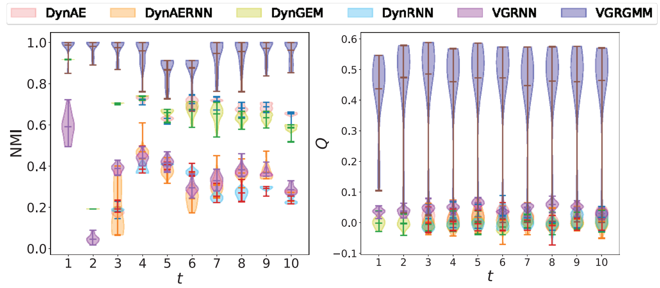


Fig. 6. Runtime comparison between VGRGMM and deep VAE-based methods. We set 30 iterations on fbmessages and Var datasets for each method, and the result shows that the average run time cost of DynRNN is the highest among these methods, meanwhile, our method has the same average runtime cost with VGRNN. Though DynAE has the smallest runtime cose, it is not good at all of the downstream tasks.

uncertainty of the network but also has the characteristics of deep models, that is, the powerful computing ability.

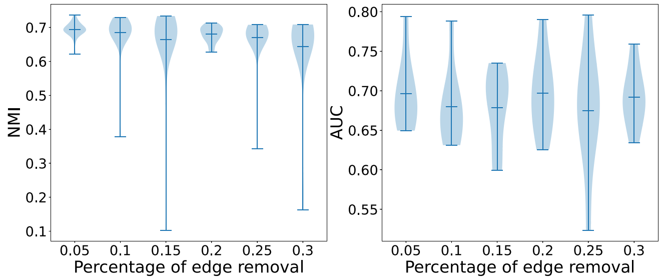
In addition, we compare the runtime of VGRGMM and DynAE, DynGEM, DynAERNN, and VGRNN on the fbmessages dataset to verify the efficiency of our method. Because these baselines are deep VAE methods, which are generally faster than traditional methods such as DYNMOGA. Furthermore, the time complexity of DynGEM and DynAE is $O(TND^2E)$, where T is the number of slices, N is the number of nodes, D is the size of the trainable graph filter, that is, the embedding dimensions, and E is the number of edges. While the time complexity of DynRNN, DynAERNN, VGRNN, and VGRGMM is $O(TND^2W)$, where W is the total parameter numbers of LSTM. Though the time complexity of these methods does not have a big gap, the realization and the optimization of them are not the same. Fig. 6 shows that the average runtime of VGRGMM is the same as VGRNN, and higher than DynGEM and DynAE. But comparing to DynRNN and considering the accuracy of downstream tasks, the efficiency of VGRGMM is acceptable.

Furthermore, the tSNE visualization of our method in Appendix A together with the case study in Appendix B also shows the embedding of nodes of our method can preserve the community information to enhance the downstream tasks, which proves that the mesoscopic information of the dynamic network can significantly affect the evolution of macroscale and the microscopic structure.



(a) (b)

Fig. 7. Violin plot on real-world dataset cellphone and synthetic dataset Var. The x-axis is the number of slices t, while the y-axis is modularity Q or NMI. Each method runs 30 times until convergence, and the result is used to generate the result probability density (the curve on each side of the violin shape), while the horizontal line in the violin is the mean value of each method on each slice. Note that DynAE, DynRNN, and DynAERNN have a look back mechanism; thus, they do not have results at $t = 1, 2$. (a) Cellphone dataset. (b) Var dataset.



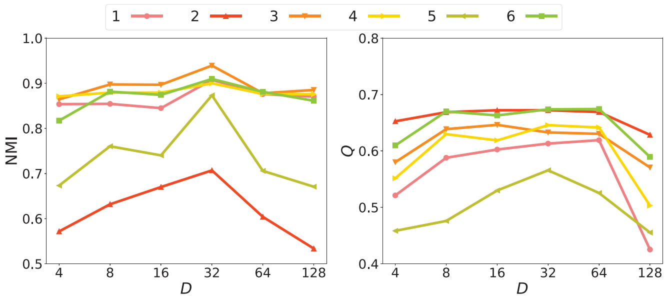
(a) (b)

Fig. 8. Violin plot on workplace dataset in terms of NMI and AUC. To test the robustness of VGRGMM, we remove the edges of workplace dataset from 5% to 30%, then run VGRGMM 30 times to obtain the result distribution. The x-axis is the percentage of edge removal, while the y-axis is the NMI or AUC result distribution. (a) NMI on cellphone dataset. (b) AUC on cellphone dataset.

E. Statistical Test and Robustness Test

To further evaluate the stability in terms of initialization and network disturbance, we also make a statistical test and a robustness test with our method and the deep VAE-based baselines. Fig. 7 shows that on both synthetic and real-world datasets, VGRGMM outperforms baseline methods despite a wider result distribution. Meanwhile, the result distribution of VGRGMM is wider than baseline, indicating that our method is sensitive to initialization in terms of effectiveness. This is a drawback of GMM distribution of our method. We can solve it by replacing GMM with Dirichlet distribution or other multimodal distributions with lower entropy.

We also make a robustness test to identify the stability of our method in terms of network disturbance. Fig. 8 shows that VGRGMM is not sensitive in terms of network disturbance. The NMI result distribution indicates that the KL divergence of our loss function (the GMM part) works well, and can accurately characterize the higher order relationships, that is, community membership. While the AUC [59] result distribution shows that the generative ability of our VAE structure, and this also indicates that our GRU item successfully characterizes the dynamic feature of the workplace dataset.



(a) (b)

Fig. 9. Parameter sensitivity analysis of VGRGMM. (a) Performance based on NMI with varying D on the Primary network. (b) Results of different embedding dimensions based on Q using Enron dataset.

F. Parameter Analysis

Fortunately, in our model VGRGMM, there are no additional balance parameters except the number of embedding dimension D. Here, we analyze the sensitivity of D to the community detection results on dynamic networks. Here, we select two datasets: 1) Primary and 2) Enron, evaluated with the NMI and Q, respectively. Just as shown in Fig. 9, we report the results of VGRGMM with varying D from 4 to 128 for each snapshot on the two networks. VGRGMM can achieve the best performance when the embedding dimension is 32. If this embedding dimension $D < 32$, there is not enough representation space to reveal the network structure. Otherwise, the embeddings will overfit the dynamic network when $D > 32$, so we set $D = 32$ in our experiments.

V. Conclusion

In this article, we proposed a deep learning-based model VGRGMM for dynamic community detection, which combines variational autoencoder with GMM and the variant of GRU to jointly capture community structure and evolution patterns. This method is capable of directly learning community detection results, network evolution, and community-aware latent representation, which is different from previous dynamic network representation methods. In particular, the latent representations are parameters of a probabilistic graphic model GMM, similar to community division. Then, it does not require a second step for clustering, such as applying k-means to the learned representation, thereby simplifying the process. We conduct extensive experiments on real-world networks and artificial networks compared with several state-of-the-art methods. The results show that our proposed model has an excellent performance in improving the accuracy of dynamic community detection, reducing time complexity, and enhancing capabilities of network representation.

VI. Limitations and Future Work

The variational autoencoder structure combined with GMM prior is a typical Bayesian deep learning paradigm, which can accurately model temporal network, thus, the parameter of our model can be used to analyze and inference network evolution mechanism. Combining with different real-world network data, our method can be used to analyze the evolution of the research team (with citation networks) [60], social network evolution analysis (with social networks) [5], and so on. But as for now, the VGRGMM still has some limitations. Currently, VGRGMM needs to specify the number of communities K in advance, which is usually unknown in real-world data. In the future, we will conduct more in-depth research on dynamic network embedding and mixed community structure on heterogeneous networks. Furthermore, we will explore other methods rather than embedding to predict dynamic events in the temporal community.

References

[1] M. E. Newman and M. Girvan, “Finding and evaluating community structure in networks,” *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 69, no. 2, 2004, Art. no. 26113.

[2] P. K. Gopalan and D. M. Blei, “Efficient discovery of overlapping communities in massive networks,” *Proc. Nat. Acad. Sci.*, vol. 110, no. 36, pp. 14534–14539, 2013.

[3] B. Chikhaoui, M. Chiazzaro, and S. Wang, “A new granger causal model for influence evolution in dynamic social networks: The case of DBLP,” in *Proc. AAAI Conf. Artif. Intell.*, vol. 29, 2015, pp. 51–57.

[4] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, “Uncovering the overlapping community structure of complex networks in nature and society,” *Nature*, vol. 435, no. 7043, pp. 814–818, 2005.

[5] J. Zhang, L. Tan, and X. Tao, “On relational learning and discovery in social networks: A survey,” *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 8, pp. 2085–2102, 2019.

[6] J. J. Whang, D. F. Gleich, and I. S. Dhillon, “Overlapping community detection using neighborhood-inflated seed expansion,” *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 5, pp. 1272–1284, May 2016.

[7] G. Rossetti and R. Cazabet, “Community discovery in dynamic networks: A survey,” *ACM Comput. Surveys*, vol. 51, no. 2, pp. 1–37, 2018.

[8] M. Del Vicario *et al.*, “The spreading of misinformation online,” *Proc. Nat. Acad. Sci.*, vol. 113, no. 3, pp. 554–559, 2016.

[9] L. Lü, L. Pan, T. Zhou, Y.-C. Zhang, and H. E. Stanley, “Toward link predictability of complex networks,” *Proc. Nat. Acad. Sci.*, vol. 112, no. 8, pp. 2325–2330, 2015.

[10] W. Yu, W. Cheng, C. C. Aggarwal, H. Chen, and W. Wang, “Link prediction with spatial and temporal consistency in dynamic networks,” in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 3343–3349.

[11] L. Ma, J. Li, Q. Lin, M. Gong, C. A. C. Coello, and Z. Ming, “Reliable link inference for network data with community structures,” *IEEE Trans. Cybern.*, vol. 49, no. 9, pp. 3347–3361, Sep. 2019.

[12] X. Tang and C. C. Yang, “Detecting social media hidden communities using dynamic stochastic blockmodel with temporal dirichlet process,” *ACM Trans. Intell. Syst. Technol.*, vol. 5, no. 2, pp. 1–21, 2014.

[13] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela, “Community structure in time-dependent, multiscale, and multiplex networks,” *Science*, vol. 328, no. 5980, pp. 876–878, 2010.

[14] F. Liu, D. Choi, L. Xie, and K. Roeder, “Global spectral clustering in dynamic networks,” *Proc. Nat. Acad. Sci.*, vol. 115, no. 5, pp. 927–932, 2018.

[15] L. Zhang, H. Pan, Y. Su, X. Zhang, and Y. Niu, “A mixed representation-based multiobjective evolutionary algorithm for overlapping community detection,” *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2703–2716, Sep. 2017.

[16] X. Zhang, K. Zhou, H. Pan, L. Zhang, X. Zeng, and Y. Jin, “A network reduction-based multiobjective evolutionary algorithm for community detection in large-scale complex networks,” *IEEE Trans. Cybern.*, vol. 50, no. 2, pp. 703–716, Feb. 2020.

[17] X. Ma and D. Dong, “Evolutionary nonnegative matrix factorization algorithms for community detection in dynamic networks,” *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 5, pp. 1045–1058, May 2017.

[18] M. Pensky and T. Zhang, “Spectral clustering in the dynamic stochastic block model,” *Electron. J. Stat.*, vol. 13, no. 1, pp. 678–709, 2019.

[19] F. Folino and C. Pizzuti, “An evolutionary multiobjective approach for community discovery in dynamic networks,” *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1838–1852, Aug. 2014.

[20] X. Zeng, W. Wang, C. Chen, and G. G. Yen, "A consensus community-based particle swarm optimization for dynamic community detection," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2502–2513, Jun. 2020.

[21] J. Sun, W. Zheng, Q. Zhang, and Z. Xu, "Graph neural network encoding for community detection in attribute networks," *IEEE Trans. Cybern.*, early access, Feb. 10, 2021, doi: 10.1109/TCYB.2021.3051021.

[22] A. Zhiyuli, X. Liang, Y. Chen, and X. Du, "Modeling large-scale dynamic social networks via node embeddings," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 10, pp. 1994–2007, Oct. 2019.

[23] Q. Liu, C. Long, J. Zhang, M. Xu, and P. Lv, "TriATNE: Tripartite adversarial training for network embeddings," *IEEE Trans. Cybern.*, early access, Mar. 17, 2021, doi: 10.1109/TCYB.2021.3061771.

[24] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE Trans. Big Data*, vol. 6, no. 1, pp. 3–28, Mar. 2020.

[25] P. Goyal, S. R. Chhetri, and A. Canedo, "Dyngraph2Vec: Capturing network dynamics using dynamic graph representation learning," *Knowl. Based Syst.*, vol. 187, Jan. 2020, Art. no. 104816.

[26] R. Trivedi, H. Dai, Y. Wang, and L. Song, "Know-evolve: Deep temporal reasoning for dynamic knowledge graphs," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3462–3471.

[27] Q. Zhang and Q. Yao, "Dynamic uncertain causality graph for knowledge representation and reasoning: Utilization of statistical data and domain knowledge in complex cases," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1637–1651, May 2018.

[28] L. Zhou, Y. Yang, X. Ren, F. Wu, and Y. Zhuang, "Dynamic network embedding by modeling triadic closure process," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, 2018, pp. 571–578.

[29] S. Cavallari, V. W. Zheng, H. Cai, K. C.-C. Chang, and E. Cambria, "Learning community embedding with community detection and node embedding on graphs," in *Proc. ACM Conf. Inf. Knowl. Manage.*, 2017, pp. 377–386.

[30] J. J. Choong, X. Liu, and T. Murata, "Learning community structure with variational autoencoder," in *Proc. IEEE Int. Conf. Data Min. (ICDM)*, 2018, pp. 69–78.

[31] W. Shi, L. Huang, C. D. Wang, J. H. Li, Y. Tang, and C. Fu, "Network embedding via community based variational autoencoder," *IEEE Access*, vol. 7, pp. 25323–25333, 2019.

[32] D. Jin, B. Li, P. Jiao, D. He, and H. Shan, "Community detection via joint graph convolutional network embedding in attribute network," in *Proc. Int. Conf. Artif. Neural Netw.*, 2019, pp. 594–606.

[33] M. Yang, J. Liu, L. Chen, Z. Zhao, X. Chen, and Y. Shen, "An advanced deep generative framework for temporal link prediction in dynamic networks," *IEEE Trans. Cybern.*, vol. 50, no. 12, pp. 4946–4957, Dec. 2020.

[34] A. Pareja *et al.*, "EvolveGCN: Evolving graph convolutional networks for dynamic graphs," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 4, 2020, pp. 5363–5370.

[35] A. e, E. Hajiramezanali, K. Narayanan, N. Duffield, M. Zhou, and X. Qian, "Variational graph recurrent neural networks," in *Advances in Neural Information Processing Systems*, vol. 32. Red Hook, NY, USA: Curran Assoc., 2019.

[36] S. M. Kazemi *et al.*, "Representation learning for dynamic graphs: A survey," *J. Mach. Learn. Res.*, vol. 21, no. 70, pp. 1–73, 2020.

[37] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. 2nd Int. Conf. Learn. Represent.*, 2014, pp. 1–14.

[38] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," 2015, *arXiv:1506.00019*.

[39] K. S. Xu, M. Kliger, and A. O. Hero, III, "Adaptive evolutionary clustering," *Data Min. Knowl. Disc.*, vol. 28, no. 2, pp. 304–336, 2014.

[40] F. Liu, J. Wu, S. Xue, C. Zhou, J. Yang, and Q. Sheng, "Detecting the evolving community structure in dynamic social networks," in *Proc. 29th Int. Conf. World Wide Web*, vol. 23, 2020, pp. 715–733.

[41] D. K. Sewell and Y. Chen, "Latent space approaches to community detection in dynamic networks," *Bayesian Anal.*, vol. 12, no. 2, pp. 351–377, 2017.

[42] L. Zhu, D. Guo, J. Yin, G. V. Steeg, and A. Galstyan, "Scalable temporal latent space inference for link prediction in dynamic social networks," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 10, pp. 2765–2777, Oct. 2016.

[43] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2016, pp. 1225–1234.

[44] P. Goyal, N. Kamra, X. He, and Y. Liu, "DynGEM: Deep embedding method for dynamic graphs," 2018, *arXiv:1805.11273*.

[45] Y. Zhao, X. Wang, H. Yang, L. Song, and J. Tang, "Large scale evolving graphs with burst detection," in *Proc. IJCAI*, 2019, pp. 4412–4418.

[46] J. Yang and J. Leskovec, "Overlapping communities explain core–periphery organization of networks," *Proc. IEEE*, vol. 102, no. 12, pp. 1892–1902, Dec. 2014.

[47] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017, pp. 1–14.

[48] T. Kipf and M. Welling, "Variational graph auto-encoders," in *Proc. NIPS Workshop Bayesian Deep Learn.*, 2016, pp. 1–3.

[49] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured sequence modeling with graph convolutional recurrent networks," in *Proc. 25th Int. Conf. Neural Inf. Process.*, 2018, pp. 362–373.

[50] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou, "Variational deep embedding: An unsupervised and generative approach to clustering," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 1965–1972.

[51] M.-S. Kim and J. Han, "A particle-and-density based evolutionary clustering method for dynamic networks," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 622–633, 2009.

[52] Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng, "Analyzing communities and their evolutions in dynamic social networks," *ACM Trans. Knowl. Disc. Data*, vol. 3, no. 2, p. 8, 2009.

[53] D. J. DiTursi, G. Ghosh, and P. Bogdanov, "Local community detection in dynamic networks," in *Proc. IEEE Int. Conf. Data Min.*, 2017, pp. 847–852.

[54] J. Stehlé *et al.*, "High-resolution measurements of face-to-face contact patterns in a primary school," *PloS One*, vol. 6, no. 8, 2011, Art. no. e23176.

[55] M. Génois, C. L. Vestergaard, J. Fournet, A. Panisson, I. Bonmarin, and A. Barrat, "Data on face-to-face contacts in an office building suggest a low-cost vaccination strategy based on community linkers," *Netw. Sci.*, vol. 3, no. 3, pp. 326–347, 2015.

[56] R. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proc. AAAI Conf. Artif. Intell.*, vol. 29, 2015, pp. 4292–4293.

[57] N. Mrabah, N. M. Khan, R. Ksantini, and Z. Lachiri, "Deep clustering with a dynamic autoencoder: From reconstruction towards centroids construction," *Neural Netw.*, vol. 130, pp. 206–228, Oct. 2020.

[58] F. Krzakala *et al.*, "Spectral redemption in clustering sparse networks," *Proc. Nat. Acad. Sci.*, vol. 110, no. 52, pp. 20935–20940, 2013.

[59] J. M. Lobo, A. Jiménez-Valverde, and R. Real, "AUC: A misleading measure of the performance of predictive distribution models," *Global Ecol. Biogeogr.*, vol. 17, no. 2, pp. 145–151, 2008.

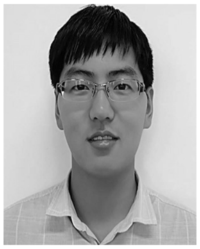
[60] L. Wu, D. Wang, and J. A. Evans, "Large teams develop and small teams disrupt science and technology," *Nature*, vol. 566, no. 7744, pp. 378–382, 2019.

Tianpeng Li received the M.S. degree from the School of Software Engineering, Tianjin University, Tianjin, China, in 2020, where he is currently pursuing the Ph.D. degree with the College of Intelligence and Computing.

His research interests include machine learning, dynamic complex network analysis, dynamic community detection, and community evolution in dynamic network and probabilistic graphical models and its applications in computer science.

Wenjun Wang received the Ph.D. degree from the School of Information Science and Technology, Peking University, Beijing, China, in 2006.

He is currently a Professor with the College of Intelligence and Computing, Tianjin University, Tianjin, China, the Chief Expert of major projects of the National Social Science Foundation, the Big Data Specially Invited Expert of Tianjin Public Security Bureau, and the Director of the Tianjin Engineering Research Center of Big Data on Public Security. His research interests include computational social science, large-scale data mining, intelligence analysis, and multilayer complex network modeling.



Pengfei Jiao (Member, IEEE) received the Ph.D. degree in computer science from Tianjin University, Tianjin, China, in 2018.

From 2018 to 2021, he was a Lecturer with the Center of Biosafety Research and Strategy, Tianjin University. He is currently a Professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His current research interests include complex network analysis and its applications.

Yinghui Wang received the master's degree from Tianjin University, Tianjin, China, in 2018, where she is currently pursuing the Eng.D. degree with the School of Computer Science and Technology.

Her current research interests include complex network analysis and computational social science.

Ruomeng Ding is currently pursuing the bachelor's degree with the School of Computer Science and Technology, Tianjin University, Tianjin, China.

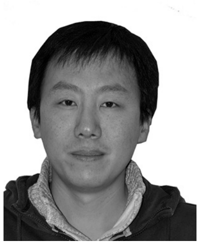
Her current research interests include complex network analysis, dynamic community detection, and network representation learning.

Huaming Wu (Senior Member, IEEE) received the B.E. and M.S. degrees in electrical engineering from the Harbin Institute of Technology, Harbin, China, in 2009 and 2011, respectively, and the Ph.D. degree (with Highest Hons.) in computer science from Freie Universität Berlin, Berlin, Germany, in 2015.

He is currently an Associate Professor with the Center for Applied Mathematics, Tianjin University, Tianjin, China. His research interests include model-based evaluation, wireless networks, edge/cloud computing, deep learning, and complex networks.

Lin Pan received the Ph.D. degree in computer science from Tianjin University, Tianjin, China, in 2016.

He is a Lecturer with the School of Marine Science and Technology, Tianjin University. His current research interests include complex network analysis, social computing, and marine data mining.



Di Jin received the Ph.D. degree in computer science from Jilin University, Changchun, China, in 2012.

He was a Postdoctoral Research Fellow with the School of Design, Engineering, and Computing, Bournemouth University, Poole, U.K., from 2013 to 2014. He is currently an Associate Professor with the College of Intelligence and Computing, Tianjin University, Tianjin, China. He has published more than 50 papers in international journals and conferences in the areas of community detection, social network analysis, and machine learning.

Dr. Jin serves as an Invited Reviewer for journals, including IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and a Senior Program Committee Member for conferences, including AAAI.