Dynamic community discovery via common subspace projection

New Journal of Physics
The open access journal at the forefront of physics

Deutsche Physikalische Gesellschaft DPG
IOP Institute of Physics

Published in partnership with: Deutsche Physikalische Gesellschaft and the Institute of Physics

OPEN ACCESS

RECEIVED
16 October 2020

REVISED
29 January 2021

ACCEPTED FOR PUBLICATION
10 February 2021

PUBLISHED
19 March 2021

PAPER

Dynamic community discovery via common subspace projection

Lanlan Yu[1], Ping Li[1,*], Jie Zhang[2,*] and Jürgen Kurths[3]

[1] Center for Intelligent and Networked Systems, School of Computer Science, Southwest Petroleum University, Chengdu 610500, People's Republic of China
[2] Institute of Science and Technology for Brain-Inspired Intelligence, Fudan University, Shanghai 200433, People's Republic of China
[3] Potsdam Institute for Climate Impact Research, Telegraphenberg A 31, 14473 Potsdam, Germany
[*] Authors to whom any correspondence should be addressed.

E-mail: dping.li@gmail.com and jzhang080@gmail.com

Keywords: dynamic networks, dynamic community detection, noise filtering

Abstract

Detecting communities of highly internal and low external interactions in dynamically evolving networks has become increasingly important owing to its wide applications in divers fields. Conventional solutions based on static community detection approaches treat each snapshot of dynamic networks independently, which may fragment communities in time (Aynaud T and Guillaume J L 2010 *8th Int. Symp. on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks* (IEEE) pp 513–9), resulting in the problem of instability. In this work, we develop a novel dynamic community detection algorithm by leveraging the encoding–decoding scheme present in a succinct network representation method to reconstruct each snapshot via a common low-dimensional subspace, which can remove non-significant links and highlight the community structures, resulting in the mitigation of community instability to a large degree. We conduct experiments on simulated data and real social networking data with ground truths (GT) and compare the proposed method with several baselines. Our method is shown to be more stable without missing communities and more effective than the baselines with competitive performance. The distribution of community size in our method is more in line with the real distribution than those of the baselines at the same time.

1. Introduction

Community detection, also known as node clustering, is one of the most active topics in the field of graph mining and network science [2–5]. The task of community detection is to group the vertices of a graph into clusters by considering the connection structure in such a way that the edges within the cluster far exceed the edges between them. Generally, the wide variety of grouping techniques are based on similarity measures defined on structural properties of the vertices [6–8]. As community detection has been used extensively in many applications, such as identifying political parties [9], genetically similar structures [6] and fraud in telecommunication networks [10], there are many community detection methods available in the literature [11, 12]. Most existing approaches are developed to tackle static community discovery, where network connections are fixed in time. However, in many real world systems, the relationships between the nodes are constantly changing, leading to the evolution of community structures. That is, a community may grow by absorbing new members or become smaller due to the leave of some of its nodes, or even disappear after a period of time. Examples include social networks such as active collaboration circles [13], video sharing [14], communication networks such as email communications [15], and food webs [16].

Though the conventional community detection methods for static networks can be applied in dynamic scenarios when a time-evolving network is represented by a sequence of snapshots of networks, a fundamental drawback rooted in such scheme is that most of traditional algorithms are sensitive to tiny changes in the network structure [1, 17–19]. Specifically, for the two similar networks, the algorithm can provide very different partition results even if a few links are disturbed. Besides, as dynamic community discovery is usually associated with community tracking [20], the framework of an independent identification of each snapshot needs an additional set matching operation to obtain the evolution trajectories of the nodes' affiliations. Instead of detecting and tracking communities in phases like the aforementioned solution, we propose to achieve the two goals simultaneously. Precisely, we tailor the succinct representation method [21] to project all snapshots into a common but lower-dimensional subspace where the snapshots can be reconstructed and compared more effectively. It should be noted that the proposed method here is different from the previous work [21]. First, that work aims at summarizing a dynamic network, i.e. finding a representation of all snapshots of the dynamic network in a period of interest, which is achieved with principal component analysis (PCA) on the multiple network snapshots. In the present method, however, we focus on dynamic community detection using node similarities across the period of interest, i.e. the correlation between nodes' concatenated connectivity profile. Although we also use PCA, this technique is applied to the node similarity matrix[4] to obtain an eigen-space, wherein all connections among nodes at each time step can be reconstructed. The benefits of this operation are, it not only makes the strong ties between similar nodes retained, but serves the need of global smoothing.

Furthermore, we compare the proposed method with various existing baselines including the state-of-the-art methods and show the superiority of our method. Surprisingly, we find that most existing dynamic community detection approaches lack adaptiveness to different community evolution patterns, which will be detailed in the experimental results. Besides, we explore the community properties related to the performance of the algorithms and especially analyze the results detected by our method.

In the remainder of this paper, we first provide a short summary of previous works relevant to this topic in section 2. In section 3 we describe the proposed method in detail, followed by its experimental evaluations and discussion. Finally, we conclude with a discussion of the proposed method and highlight some promising directions for future work.

2. Related work

2.1. Static community discovery

A community is described as a substructure in a network [22], where the nodes are more densely connected internally than with the rest of the network. However, community is not defined strictly, but algorithmically [12]. Even so, it is still of particular importance to view the network structure in mesoscales. To this end, many approaches have been developed to mine the communities in modular networks [23–25]. Generally, those approaches fall into several categories: node similarity based method [26], optimization oriented method [27], network dynamics based method [28] and statistical modeling [25]. Among various methods, spectral clustering [29] is a commonly used community detection method based on nodes' similarity, which has been adopted to solve the problem of overlapping community structure [7]. Spectral clustering based community detection intimately associates with subspace sparse coding. Several recent studies [8, 30, 31] have shown that, by introducing subspace sparse representations of the nodes, the performance of spectral clustering can be improved.

However, most of the existing community discovery algorithms work on static networks without temporal information.

2.2. Dynamic community discovery

Community discovery in dynamic networks focuses on identifying and tracing the community evolution, which can characterize the time-varying behaviors of the dynamic network as well [18, 32]. Generally, the dynamic community discovery algorithms fall into two main categories: one-stage methods [33, 34] and two-stage methods [1, 35]. A one-stage method is designed by detecting and tracing simultaneously, TimeRank [34] falls in this category. By labeling the time attribution on the nodes of each snapshot, and transferring the dynamic network into a static network, TimeRank method [34] uses the static community discovery algorithm to detect the cross-time communities and trace the evolution of communities. In contrast, a two-stage method detects communities in each snapshot and then matches each community between pairs of snapshots to trace the dynamic community structure. Moreover, the two-stage method is more complex than the one-stage method due to additional matching for all potential community pairs. A critical issue in one-stage dynamic community detection is the stability of the results, which generally involves two aspects: a tiny variation in network structure may lead to quite different detection results for a generic algorithm, and the algorithms may fail to detect meaningful communities due to the complex

[4] Node similarity matrix is used to organize the mutual similarities between nodes, where the similarity is measured based on the common historical neighbors of two nodes.

evolution processes of dynamic networks [1]. Note that, the two-stage method still cannot solve the instability problem if its detecting operation does not involve the smoothing over snapshots [18, 32]. Recently, by considering the time-dependent relationship between sequential snapshots, some two-stage methods [36, 37] have shown its advances in mitigating the instability. Those methods belong to the evolutionary clustering [19].

In this work, we propose a one-stage method. But different from the existing algorithms, our approach solves the problem from a signal processing perspective, i.e. we treat a dynamic network as the combination of signal connections and noisy connections, for which the theoretical basis is demonstrated in the following section. We first generate a similarity matrix to characterize the proximity strengths of the nodes in terms of their dynamic connection records. By leveraging the above signal combination assumption, we suppose that small weights in similarity matrix may indicate noisy links. There we use the PCA technique on similarity matrix to get the signal subspace, where the links between dissimilar nodes can be filtered out by projection and reconstruction operations on the connections of the original network snapshots. As a result, the connections between different communities are most likely to be removed and the relationships among nodes in the communities are emphasized, facilitating the identification of communities. Compared to two-stage methods that require additional matching for all potential community pairs, our method detects the communities all at once, which makes it more efficient than two-stage methods. Moreover, since our method is performed on the snapshots of dynamic networks, not the conjunction of all snapshots in many one-stage methods, our method is also superior to the state-of-the-art one-stage method such as TimeRank.

3. Problem formulation

Considering a dynamic network G that describes the evolution of relations between interacting objects, there are two ways to model it: by temporal networks [38–40] or alternatively, by a sequence of snapshots [41]. Our proposed method belongs to the latter form. That is, $G = \{G_t | t = 1, 2, \dots, T\}$, where G_t is the snapshot at the tth time, and T is the length of the sequence G. Let A_t denote the adjacency matrix corresponding to the snapshot $G_t = (V_t, E_t)$, where E_t and V_t are the edges and nodes appearing in G_t, respectively. Accordingly, the node set of G is $V = \cup V_t$ within the period-of-interest.

Dynamic community discovery aims to detect all dynamic communities in a dynamic network. We define the dynamic communities $C = \{C_1, \dots, C_j, \dots, C_K\}$ as a set of clusters with constant labels C_j ($j = 1, \dots, K$), where K is the total number of dynamic communities. Moreover, $C_j = \{i_t | i_t \in V_t, 1 \leqslant t \leqslant T\}$ consists of nodes from different time steps, where i_t denotes node i at time t. That is, the community members are time-stamped. As such, it is convenient to observe the life-cycle of a dynamic community as well as the transition of community affiliation of a node [32]. For instance, to see how many members a community C_j has at a specific time t_0, one can count the nodes with time stamp t_0 in C_j. Similarly, since there is a correspondence between community affiliations and time-stamped nodes, by tracing the community affiliation of a node in time, the transition from one affiliation to another can be detected.

Generally, discovering the underlying community patterns in networks is also known as clustering of nodes, which is usually based on certain similarity measures defined over networks in a high-dimensional feature space. Among the existing approaches, matrix factorization has been actively used in mapping the nodes to a lower-dimensional vector subspace of the latent feature space [30, 35]. For instance, the spectral decomposition of the Laplacian of the adjacency matrix or its variants that can embed nodes into the space composed of one or more eigenvectors is a common practice. In this work, the proposed method is also based on matrix factorization. However, different from spectral clustering where the nodes are encoded in a subspace and then spatial clustering algorithm is applied on the embedding, we construct a common subspace to act as the filter for the connection among nodes, which can be derived from the following theoretical analysis.

Since the connections that link different communities usually cause the ambiguity of community structure and affect the decision boundary of detection models, such connections can be treated as noise. From this point of view, we decompose the zero-mean similarity matrix as $\bar{M} = X + W$, where X denotes the signal part and W is the noise part. Suppose that all signals and noises are mutually uncorrelated and the noises have identical variance δ, then the covariance matrix $R = \bar{M}^{\mathrm{T}} \times \bar{M}$ can be rewritten as

$$\bar{M}^{\mathrm{T}} \times \bar{M} = X^{\mathrm{T}} \times X + \delta^2 \times I, \tag{1}$$

where I is the identity matrix. By applying eigen-decomposition on R, then we have $R = U \times \Sigma \times U^{\mathrm{T}}$, where U is the eigen-matrix composed of the eigenvectors corresponding to R. Let Σ be the diagonal matrix

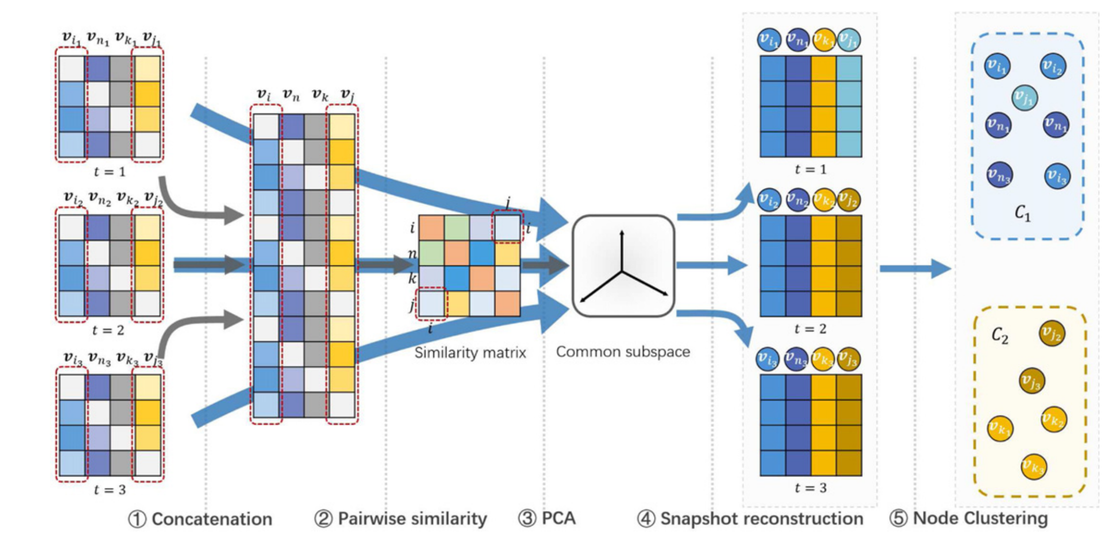


Figure 1. The framework of the proposed method. It consists of five steps: (1) concatenating nodes' instant connectivity profiles, (2) similarity measuring based on node connectivity profiles, (3) eigen-decomposition using PCA, (4) snapshot reconstruction in the derived common subspace and (5) time-stamped node clustering. There are two data processing channels indicated by gray arrows and blue arrows, respectively. Specifically, gray arrows represent the process of finding the common subspace (step (1)–(3)), while blue arrows represent the process of structural filtering on individual network snapshots (step (4), (5)).

composed of eigenvalues corresponding to R, and U be the matrix of eigenvectors, then R can be rewritten as:

$$R = \begin{bmatrix} U_s & U_n \end{bmatrix} \times \begin{bmatrix} \Sigma_s & 0 \\ 0 & \Sigma_n \end{bmatrix} \times \begin{bmatrix} U_s^{\mathrm{T}} \\ U_n^{\mathrm{T}} \end{bmatrix} \tag{2}$$

$$= U_s \times \Sigma_s \times U_s^{\mathrm{T}} + U_n \times \Sigma_n \times U_n^{\mathrm{T}}. \tag{3}$$

Suppose the signal X is low-rank and the signal-noise ratio is large, then equation (2) can be rewritten as:

$$R = U \times \Lambda \times U^{\mathrm{T}} + \delta^2 \times I, \tag{4}$$

where $\Lambda = \begin{bmatrix} U_s & 0 \\ 0 & 0 \end{bmatrix}$. Obviously, the first term in equation (2) is equivalent to the reconstructed signal matrix X. The principal components in U can be obtained by performing PCA on R.

In the light of the theoretical analysis, we use PCA to obtain a common low-dimensional subspace for all network snapshots, where the encoding–decoding operation acts as filtering and retains the strong ties between similar nodes in each snapshot. Then, we perform a conventional clustering method on the node column vectors corresponding to the decoded snapshots. As a result, changes of the community affiliations of a node can be easily identified.

4. Community discovery in common subspace

The framework of our method is diagrammatically illustrated in figure 1 with a toy example. Specifically, we first measure global similarities between nodes by comparing their connectivity profiles across time. In the example shown in the figure, the inner product is performed on each pair of historical connectivity profiles of the nodes (e.g., i, n, k and j) obtained by concatenating the adjacency matrices of snapshots, resulting in the similarity matrix (as shown in the left-hand side of figure 1. Then we employ the PCA method on the node similarity matrix to derive an eigen-subspace (as shown in the center of the plot) with which different network snapshots are reconstructed in an encoding–decoding scheme. Thus the left-hand side adjacency matrices in figure 1 can be projected to the eigen-subspace and then reconstructed with inverse projection. This way, the weak ties lying between two communities are most likely to be filtered out as noises. As a result, two nodes in the same community are much more likely similar to each other but distinct if they belong to different communities. That is, the community structure in the snapshots is more prominent, which will facilitate the follow-up clustering. Below are the details:

As the connections on each node is time-varying, a matrix $\mathbb{A} \in \mathbb{R}^{(T\times|V|)\times|V|}$ is constructed to store the historical edges in the dynamic network G. Here, $\mathbb{A}(i * t, j) = 1$ indicates that node i is connected with node j in the tth snapshot, where $i, j \in V$ and $t = \{1, 2, \ldots, T\}$. Clearly, a node can be characterized by its immediate and higher-order neighborhoods. For simplicity, we use the nearest neighbors to describe a node, that is, the ith column (or row) of $A_t(:,i)$ is a basic representation of node i in snapshot G_t, which is denoted by $\boldsymbol{v}_{i_t}$. Then the complete representation of node i is the column vector $\mathbb{A}(:,i)$, i.e. $\boldsymbol{v}_i = [{\boldsymbol{v}_{i_1}}^T, {\boldsymbol{v}_{i_2}}^T, {\boldsymbol{v}_{i_3}}^T]^{\mathrm{T}}$ as shown in figure 1.

Next, node similarity is calculated according to the similarity of connectivity profiles between two nodes. In general, the direct connections and the second order neighbors are used to depict the connectivity profile of a node, as a node is more relevant to its nearest neighbors than distant nodes, while the second order neighbors can account for the similarity when the nearest neighbors change in time frequently. In this case, the corresponding similarity matrix M can be written as follows:

$$M = \mathbb{A}^{\mathrm{T}} \times \mathbb{A}, \tag{5}$$

where $M(i,j) = \boldsymbol{v}_i^T \cdot \boldsymbol{v}_j$, is the total number of common neighbors of two nodes in T time stamps, indicating a rough similarity between nodes. As aforementioned, the direct connectivity is a straightforward indicator of the pairwise similarity, especially for the case where dynamic networks change unevenly. Thus alternatively one can combine these two components together, which results in

$$M(i,j) = \alpha \sum_{t=1}^{T} \mathbb{A}(i * t, j) + \beta(\mathbb{A}(:,i)^T \cdot \mathbb{A}(:,j)), \tag{6}$$

where $\alpha, \beta \in [0,1]$ determine the relative importance of the direct and indirect connectivity in measuring dynamic similarity between nodes, respectively. [5] Then the columns of M are employed to represent the nodes, instead of $\mathbb{A}$. Assuming that the contribution of different neighboring nodes to the characterization of a target node is different, we expect that the nodes can be expressed by their similar neighbors. Therefore, we take the links between a node pair with a low similarity value (measured by the number of common neighbors) as noise. Then, a denoising technique can be devised to derive a low-dimensional subspace based on the previous theoretical analysis.

We apply PCA to the covariance matrix M to get the principal matrix P which usually spans a low-dimensional space. The spanned space is taken as the common projection subspace. Specially, we first normalize the column representation vectors $M(:,i)$ as follows:

$$\bar{M}(:,i) = M(:,i) - \frac{1}{|V|}\sum_{j=1}^{|V|} M(:,j). \tag{7}$$

In step 2, we calculate the covariance matrix Φ,

$$\Phi = \frac{1}{|V|}\bar{M} \times \bar{M}^{\mathrm{T}}. \tag{8}$$

Step 3 performs the spectral decomposition to get the principal matrix $P = (\boldsymbol{p}_1^T, \boldsymbol{p}_2^T, \ldots, \boldsymbol{p}_s^T)^{\mathrm{T}}$, where $\boldsymbol{p}_i = (p_{i1}, \ldots, p_{i|V|}) \in \mathbb{R}^{|V|}$ is an eigenvector corresponding to the top-s eigenvalues. Generally, s is *a prior* parameter and can be set according to the distribution of the eigenvalues, which is detailed in the appendix A. Alternatively, s can also be determined by the contribution rate of s principal components, i.e. $\frac{\sum_{i=1}^{s}\lambda_i}{\sum_{j=1}^{N}\lambda_j}$. The subspace spanned by the principal matrix P is what we seek.

To mitigate the instability of the communities detected by static community detection approaches, we project all snapshots to the common subspace and then reconstruct the connection patterns by filtering the negative weights. The details are shown in the following:

(i) Firstly, the feature representations of the nodes in each snapshot are normalized to have zero-mean:

$$\bar{\boldsymbol{v}}_{i_t} = \boldsymbol{v}_{i_t} - \frac{1}{|V|}\sum_{j_t \in V_t} \boldsymbol{v}_{j_t}, \tag{9}$$

where j_t is the node in V_t and $\boldsymbol{v}_{j_t}$ is its vector representation.

(ii) Then the normalized vector $\bar{\boldsymbol{v}}_{i_t}$ is projected to the common subspace:

$$\pi_{i_t} = P \times \bar{\boldsymbol{v}}_{i_t}. \tag{10}$$

[5] In practical applications, for dynamic networks with drastic structure changes, the direct connectivity need to be more weighted than the second order one, i.e. $\alpha > \beta$, while for steady networks equal weights are recommended.

Algorithm 1. Dynamic community discovery by a common subspace (ComSP).

```
Input: $G = \{G_t | t = 1, 2, \ldots, T\}$, the dynamic network; $s$, the selected number of principal components; $K$, the number of clusters
Output: the set of dynamic communities $C$
1: $\mathbb{A} \leftarrow$ the matrix storing the historical edges in $G$
2: $M \leftarrow$ the similarity matrix produced by equation (6)
3: $P \leftarrow$ the principal matrix by equations (7) and (8)
4: $R = \oslash$
5: for $t = 1 \rightarrow T$ do
6:   for $i \in V_t$ do
7:     $\bar{\boldsymbol{v}}_{i_t} \leftarrow$ the normalized vector by equation (9)
8:     $\tilde{\boldsymbol{v}}_{i_t} \leftarrow$ the representation by equations (10)–(12)
9:     $R = R \cup \tilde{\boldsymbol{v}}_{i_t}$
10:  end for
11: end for $C \leftarrow$ the result of the $K$-means algorithm for $R$
Return $C$
```

Table 1. Statistics of the datasets: the total number of nodes (n), the minimum number of nodes and edges in the snapshots ($n_{\min}$ and $m_{\min}$, respectively), the maximum number of nodes and edges in the snapshots ($n_{\max}$ and $m_{\max}$, respectively), the average ($\bar{n}$ and $\bar{m}$) and the variance (n_{var} and m_{var}) of node and edge numbers, respectively, the number of dynamic communities (K) and the size of dynamic network (T). Decimal part is omitted.

	n	$n_{\min}$	$n_{\max}$	$\bar{n}$	n_{var}	$m_{\min}$	$m_{\max}$	$\bar{m}$	m_{var}	K	T
SBM	1000	1000	1000	1000	0	75 367	75 442	75 423	1027	4	4
Reddit-I(a)	282	117	150	128	173	130	191	150	593	3	4
Reddit-I(b)	470	114	150	129	217	112	191	144	517	3	8
Reddit-II	18	6	12	9	6	7	23	13	51	2	3
Reddit-III(a)	275	31	133	86	1666	25	176	110	3511	17	4
Reddit-III(b)	339	31	133	64	1360	25	176	74	3135	17	8

(iii) Furthermore, the low-dimensional embedding of each snapshot is decoded in the following way:

$$f_{i_t} = P^{\mathrm{T}} \times \pi_{i_t}, \qquad (11)$$

which means that the relational patterns among the nodes in each snapshot are reconstructed. In fact, $\boldsymbol{v}_{i_t}$ embodies the information about neighbors of node i. Thus, f_{i_t} reconstructs the relationship between node i and other nodes in snapshot G_t, in which positive values correspond to statistically strong connections among similar nodes but negative values indicate a weak relation over the time window. Through the above *encoding–decoding* process, f_{i_t} combines the local connection pattern with temporal information for global-smoothing.

(iv) Filtering the negative connection, we keep the strong connection for the reconstructed relationship, which is the succinct and global-smoothing representation of each node i at time t. The detailed operation is as follows:

$$\tilde{\boldsymbol{v}}_{i_t} = \delta(f_{i_t}), \qquad (12)$$

where $\delta(\cdot)$ is the step function which takes 1 for positive variables and 0, otherwise. $\tilde{\boldsymbol{v}}_{i_t}$ is the normalized vector of node i in snapshot G_t.

Finally, we cluster all the novel time-stamped representation of the nodes $\{\tilde{\boldsymbol{v}}_{i_t} | t = 1, 2, \ldots, T, i = 1, 2, \ldots, |V|\}$ using existing clustering methods such as the K-means algorithm [42]. This way, detecting and tracing dynamic communities are done in one stage. Specifically, the label of clusters are constant over time in this case and the nodes that appear in several snapshots may have multiple community labels, which makes it very convenient to trace a possible change of each community members and the evolution of communities as well.

Algorithm 1 gives a detailed description of the whole procedure, which is called ComSP for short. We find that the time complexity of the algorithm is of $O(T \times |V|^2 \times K)$, which is lower than the state-of-the-art one-stage method TimeRank [34].

5. Experiments

5.1. Dataset description

We conduct experiments using synthetic networks and a real-world social network with different periods. Table 1 summarizes the statistics of the datasets.

Synthetic networks is a sequence networks with four snapshots, designed to imitate the evolution of communities in dynamic networks. We use the stochastic block model (SBM) [43–47] to generate four snapshots, where the cross-block connection probability is set to 0.2 and the in-block connection probability is 0.6. Moreover, there are 4 communities with 250 nodes for each in these snapshots. Then a perturbation is applied on the edges of the snapshots with the probability 0.01. Particularly, in the last snapshot, the community affiliations of two randomly selected nodes are changed, compared to the previous snapshot. Hence, this sequence embodies typical operations on communities, i.e. apparent continuation and unnoticeable growth/contraction [48].

Reddit[6] the raw dataset records the social networking activities with the threads of posts and comments associated with subreddits, e.g. 'dogs', 'tennis', which are the innate community labels for the networks composed of the users and their interactions. We take these subreddits as the ground truths (GT) of communities. Moreover, the community label of a node is determined by the majority of the subreddits their replies belong to. Then the communities of the evolving social network behave in a dynamic way, e.g. birth/death, growth/contraction and merging/splitting. Note that, to reduce the influence of random sampling on the performance comparison, we use the entire dataset contributed by Pushshift[7]. Based on the raw records, we construct three sequences of weekly unweighted and undirected networks (referred to as Reddit-I(a), Reddit-II and Reddit-III(a), respectively) from 1st September 2010 to 28th September 2010, and two longer sequences (Reddit-I(b), Reddit-III(b)) from September and October 2010. The details of how to construct these network sequences can be found in https://github.com/NightmareNyx/CommunityTracking.

5.2. Baselines and experimental settings

The baseline methods used for comparison cover all three categories of dynamic community discovery approaches [32], namely, the instant optimal that considers communities of each snapshot independently, the temporal trade-off in which communities at t is the result of trade-off between the optimal solution at t and the known past (or global optimization), and the cross-time method that searches partition solutions for all snapshots simultaneously.

MultiGL [41, 49] utilizes a multislice generalization of modularity to study the community structure of dynamic network. Specially, this method couples between successive snapshots and rewrite the weight of intra-slice and inter-slice connections to realize 'cross-time' category. Then GenLouvain algorithm with the specified quality function is used to detect communities in the new network with $|V| \times T$ nodes.

TimeRank [34] also belongs to the 'cross-time' category. In the method, a time-weighted network with $(|V| \times T)$ nodes is producted by MutuRank [50] then undergoes spectral clustering. It has two variants, i.e. TR-AOC, TR-NOC, relying on the type of relations between nodes. Since it was shown that TR-NOC is generally better than TR-AOC [34], in our experiment, we use TR-NOC as the baseline.

Spectral clustering [29] is designed for a static community discovery. Here, we use it as an 'instant optimization' method. Thus we implement the two-stage detection with spectral clustering, which is referred to as **ts-Spect** approach. Specifically, we use the spectral clustering algorithm to discover communities in each snapshot, and match the communities in two consecutive snapshots. The match quality is measured with the Jaccard index.

GDG [30] considers the continuous density field to map each node into the geometric space, then detects static community by clustering algorithm. In order to conduct comparisons with two-stage methods, we use this algorithm to detect communities in each snapshot and then align communities to trace dynamic communities.

PisCES [35] is a temporal trade-off approach, which achieves the global smoothing of the eigenvectors of each snapshot by the regularization optimization method. PisCES detects the clusters in each snapshot by applying the K-means algorithm on their corresponding smoothed eigenvectors. These two operations are performed simultaneously. Finally, we utilize the maximum matching degree to trace the community detected by PisCES.

sE-NMF [36] uses evolutionary non-negative matrix factorization as a temporal smoothness framework for community detection. Then, by the greedy search procedure, sE-NMF uses mutual information (MI) to measure similarity between clusters of successive snapshots and maps local cluster to dynamic community for tracing.

CCPSO [37] formulates dynamic community detection as multi-objective optimization problem. It regards the common knowledge between the optimal partitions of the current and previous snapshots as consensus community, then utilizes consensus community to guide particle swarm optimization (PSO) for

[6] https://files.pushshift.io/reddit/comments/.
[7] https://pushshift.io/.

Table 2. Performance comparison of the proposed method (ComSP) and nine baselines. The top-2 performance is highlighted in bold.

Dataset	Metric	ts-Spect	GDG	PisCES	sE-NMF	MultiGL	CCPSO	TR-NOC	Adj-Mat	SuRep	ComSP
SBM	K	4	4	4	4	4	2	4	4	4	4
	NMI	1.000	1.000	1.000	1.000	*0.000*	0.164	1.000	1.000	0.997	0.997
	ARI	1.000	1.000	1.000	1.000	*-0.001*	*0.017*	1.000	1.000	0.999	0.999
	P	1.000	1.000	1.000	1.000	0.250	0.300	1.000	1.000	0.999	0.999
	R	1.000	1.000	1.000	1.000	0.250	0.906	1.000	1.000	0.999	0.999
	F_1	1.000	1.000	1.000	1.000	0.250	0.451	1.000	1.000	0.999	0.999
Reddit-II	K	3	5	3	4	2	2	2	2	2	2
	NMI	0.269	0.674	0.884	0.662	0.224	**1.000**	**1.000**	0.160	**1.000**	**1.000**
	ARI	*0.078*	0.561	0.880	0.394	0.276	**1.000**	**1.000**	*-0.078*	**1.000**	**1.000**
	P	0.770	**1.000**	**1.000**	**1.000**	0.745	**1.000**	**1.000**	0.696	**1.000**	**1.000**
	R	0.556	0.635	0.929	0.5134	0.656	**1.000**	**1.000**	0.633	**1.000**	**1.000**
	F_1	0.645	0.777	0.963	0.678	0.698	**1.000**	**1.000**	0.663	**1.000**	**1.000**
Reddit-I(a)	K	5	28	1	4	8	37	3	3	3	3
	NMI	0.268	0.378	*0.000*	0.191	*0.015*	0.581	**0.710**	*0.099*	0.629	**0.719**
	ARI	0.328	0.370	*0.000*	0.123	*-0.023*	0.361	0.632	*-0.084*	**0.654**	**0.770**
	P	0.607	0.769	0.439	0.559	0.446	**0.977**	0.750	0.470	0.815	**0.860**
	R	0.666	0.346	**1.000**	0.377	0.395	0.324	**0.977**	0.817	0.763	0.829
	F_1	0.635	0.477	0.611	0.450	0.419	0.487	**0.849**	0.597	0.788	**0.844**
Reddit-I(b)	K	8	57	1	7	10	59	3	3	3	3
	NMI	0.128	0.274	*0.000*	0.179	*0.011*	0.514	*0.032*	*0.090*	**0.558**	**0.691**
	ARI	0.146	0.240	*0.000*	0.164	*-0.010*	0.291	*0.004*	*-0.043*	**0.679**	**0.747**
	P	0.493	0.695	0.410	0.556	0.416	**0.940**	0.414	0.436	0.744	**0.826**
	R	0.646	0.248	**1.000**	0.319	0.365	0.236	**0.990**	0.876	0.711	0.807
	F_1	0.559	0.366	0.582	0.405	0.389	0.377	0.584	0.582	**0.727**	**0.817**
Reddit-III(a)	K	28	44	11	54	30	39	17	17	17	17
	NMI	0.510	0.595	0.291	0.641	0.188	**0.805**	**0.785**	0.522	0.741	0.773
	ARI	0.197	0.258	*0.023*	0.217	*-0.003*	**0.596**	0.500	0.102	0.512	**0.546**
	P	0.408	0.534	0.175	0.626	0.153	**0.805**	0.615	0.420	0.617	**0.651**
	R	0.346	0.367	**0.897**	0.269	0.158	0.591	**0.798**	0.680	0.608	0.683
	F_1	0.374	0.435	0.293	0.377	0.156	**0.681**	**0.695**	0.520	0.612	0.667
Reddit-III(b)	K	36	70	34	81	39	45	17	17	17	17
	NMI	0.442	0.596	0.258	0.625	0.155	0.723	**0.754**	0.487	0.657	**0.728**
	ARI	0.140	0.222	*-0.018*	0.169	*-0.006*	0.476	**0.610**	*0.058*	0.409	**0.539**
	P	0.396	0.630	0.249	**0.693**	0.161	**0.746**	0.619	0.440	0.567	0.658
	R	0.237	0.289	0.575	0.206	0.128	0.432	**0.797**	**0.622**	0.519	0.575
	F_1	0.297	0.396	0.347	0.317	0.143	0.547	**0.696**	0.516	0.542	**0.614**

the current snapshot. At the last step, CCPSO detects the clusters for each snapshot. In this sense, CCPSO falls into the category of temporal trade-off approach. Due to lack of tracing method in its original algorithm [37], we track a community in a way similar to PisCES.

Adj-Mat uses the adjacency matrix $A_t(:, i)$ as the representations of the nodes in each snapshot, which is the basic representation $\boldsymbol{v}_{i_t}$ of our method. By performing K-means on all basic representations of each snapshot, Adj-Mat detects and traces dynamic communities.

SuRep [21] obtains the symmetric matrix M by averaging all snapshots. Then, similar to the proposed method, SuRep groups the novel time-stamped representation of the existing nodes and traces the dynamic communities.

Other methods such as non-negative tensor factorization [33] have been compared by Sarantopoul *et al* [34], and TimeRank outperformed this method in most cases. We omit it in our experiments.

5.3. Evaluation metrics

We evaluate the model performance with several metrics that are borrowed from the field of clustering and commonly used in testing and comparing methods. They are normalized mutual information (NMI) [51], averaged rand index (ARI) [52], and the BCubed version of precision (P) recall (R) and the combined metric (F_1) [53], whose definitions and calculations are detailed in the appendix B. Particularly, NMI derives from entropy in information theory, which calculates mutual information between ground truth labels and labels from clustering results. However, the most important information we can get from clustering results is not labels, but rather which nodes are clustered together and which ones are not, motivating people to define rand index. To notice the importance of small clusters in imbalanced data, ARI is proposed. Difference from the macroscopic measure of NMI and ARI, BCubed measures the precision and recall of the predicted community affiliation of each node and averages them to compare predicted communities with GT communities.

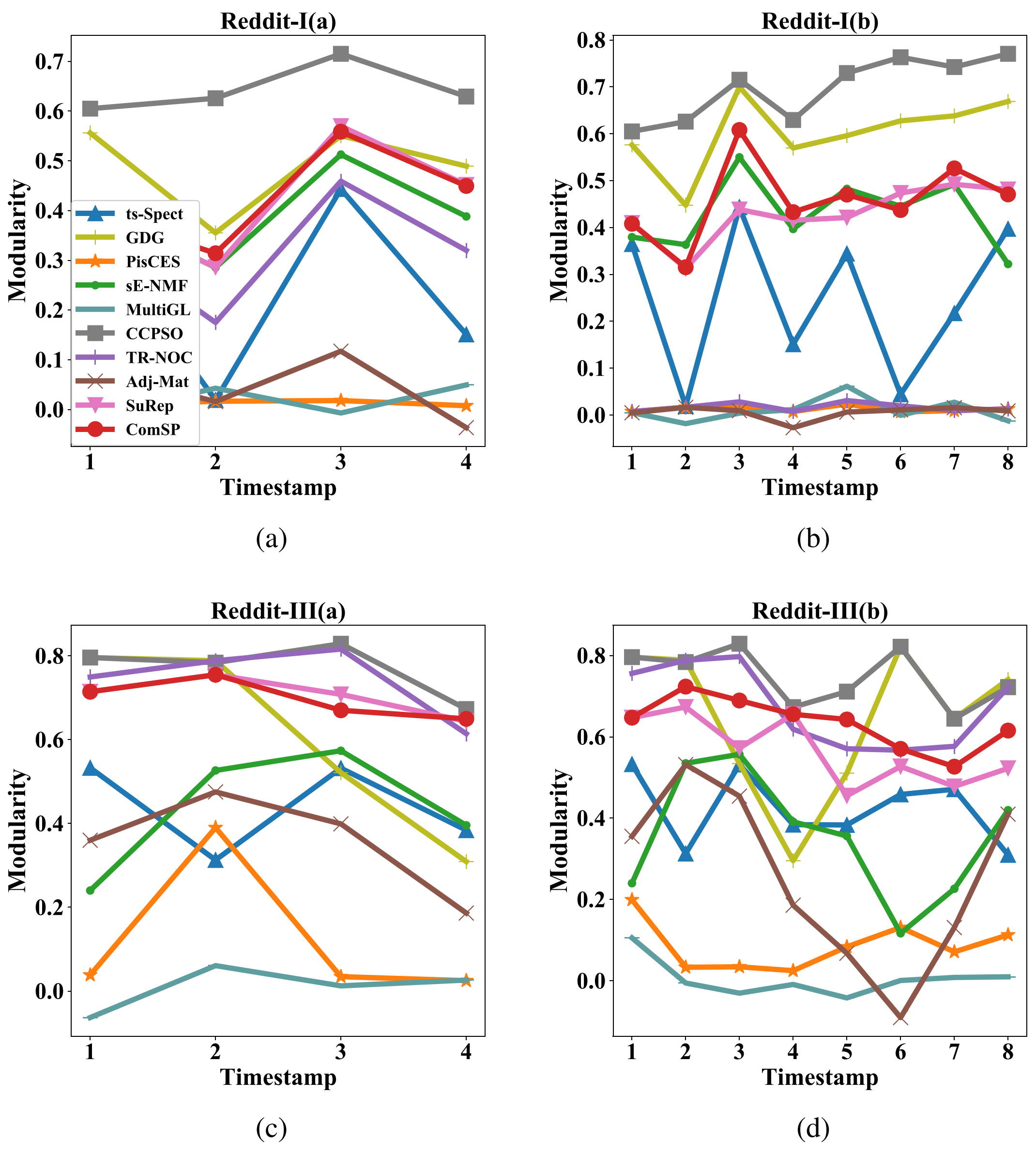


Figure 2. Modularity of community partitions on Reddit-I (subplots (a)–(b)) and Reddit-III(subplots (c)–(d)), including different community label from the ground truth and discovery dynamic communities detected by the methods TR-NOC, PisCES, ts-Spect, GDG, sE-NMF, CCPSO, Adj-Mat, SuRep and ours.

6. Results

Table 2 displays the NMI, ARI and Bcubed indexes for all six datasets. There are several observations in the results. Apparently, no method excels in every dataset. However, our proposed model shows a competitive performance, compared to the baselines. In particular, the improvement is prominent in ARI and F_1. More importantly, it is shown that the proposed method is more stable than all nine baselines when confronted with different datasets. Surprisingly, TR-NOC, PisCES and Adj-Mat identify only one giant community in Reddit-I(a) (with 4 snapshots) and Reddit-I(b) (with 8 snapshots) respectively, which is attributed to the value 0 of ARI. Similar results are found for CCPSO in SBM, implying a lack of adaptiveness in diverse evolution patterns of communities. In fact, the poor detection results of CCPSO in SBM can be traced to the its initialization (PGLP) [54], which is more likely to identify the whole network as one community in the networks where there are many links between communities, resulting in high recall but low precision and NMI index. Another observation is that, MultiGL has poor performances for all six datasets. The reason is that MultiGL take little notice of time coherence between snapshots, leading to high sensitivity to small changes of links. Moreover, our method is superior to SuRep on most of the datasets, though both of the two methods use PCA for network reconstruction. We also note that in the simulated dynamic network dataset (SBM), the proposed method and SuRep fail to spot two nodes whose community affiliations are

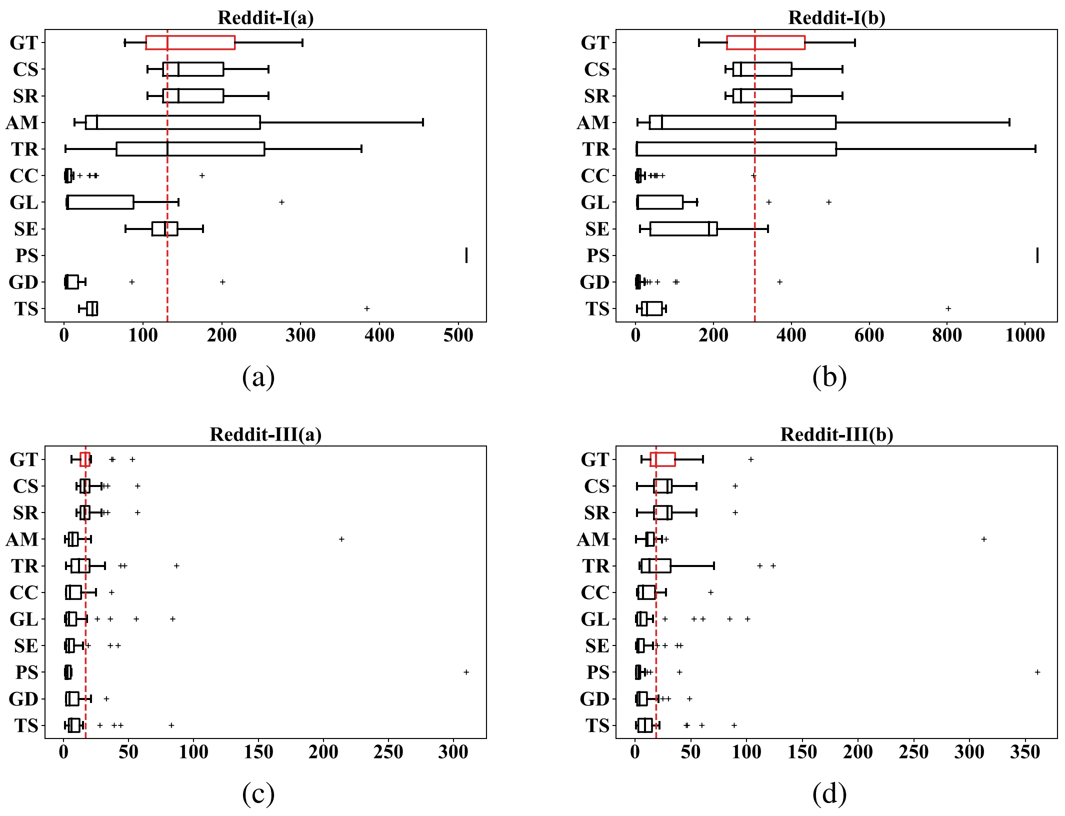


Figure 3. Community size distribution of the ground truths (GT) and discovered communities in Reddit-I (subplots (a)–(b)) and Reddit-III (subplots (c)–(d)) respectively by ten methods, i.e. TR-NOC (TR), PisCES (PS), ts-Spect (TS), GDG (GD), sE-NMF (SE), CCPSO (CC), MultiGL(GL), Adj-Mat (AM), SuRep (SR) and ComSP (CS). Red box indicates the ground truth distribution and red dotted lines represent the median of ground truth distribution.

changing in the fourth snapshot. We believe this is due to the denoising effect of the PCA operation, as trivial changes of a community will be neglected in the encoding–decoding process.

Modularity is the most widely used measure to evaluate the compactness and topological consistency of communities. However, a major drawback of using such golden quality function is that it will favor methods that are designed to maximize it, which may result in misleading comparisons. The stability of the detection methods can also be manifested from the variations of the modularity [55] at each time step, as shown in figure 2. By comparing the modularity curves, it is shown that the instant(static) community detection methods (ts-Spect and GDG) are considerably volatile with large variations of modularity, while our method is relatively more stable than the baselines in term of modularity changes and scores as well. Specifically, our method outperforms the baseline methods significantly except CCPSO and TR-NOC that are competitive on last two datasets. It should be noted that PisCES and CCPSO are designed for smoothing [32, 35, 37], leading to even curves in all datasets, especially in Reddit-III(b) there is a remarkable structural change in the snapshot at time step 5, compared to the previous snapshots, where the performance of other methods vary evidently. However, global smoothing (e.g. PisCES) overlooks the evolution of communities, which usually causes poor performance of community identification.

Besides the ground-truth based metrics and golden standard indices, the partition quality can also be evaluated by comparing the distribution of the community size resulting from the different methods and ground-truths. Figure 3 shows the sizes of communities in the boxes, where red boxes are the ground-truth distributions. Compared with the others, our method and SuRep produce the size distributions that closely approximate the ground-truth distribution, while as shown in table 2 our method excels SuRep. Moreover, the distribution of community sizes in TR-NOC shows large variances and low medians, indicating that TR-NOC tends to generate large communities, which explains why TR-NOC has high recalls. By contrast, the distribution in CCPSO has low variances and low medians, which means that it generate a lot of small communities shown in table 2 and explains why it has high precision.

Note that, the tracking operation in two-stage methods, e.g. GDG and ts-Spect, may produce lots of communities. We evaluate our method and two-stage methods by averaging the metrics of each snapshot in Reddit, the results are shown in table 3. It can be found that the proposed method shows a competitive performance, compared to the state-of-the-art two stage methods. In particular, although CCPSO is

Table 3. Performance comparisons between ComSP and the two-stage methods on the identification of community affiliations of each snapshot in Reddit. The top 2 performance is highlighted in bold, and the second-best results are also underlined.

	Reddit-I(a)					Reddit-I(b)					Reddit-III(a)					Reddit-III(b)				
Method	NMI	ARI	P	R	F_1	NMI	ARI	P	R	F_1	NMI	ARI	P	R	F_1	NMI	ARI	P	R	F_1
ts-Spect	0.338	0.331	0.642	0.778	0.700	0.272	0.213	0.566	0.767	0.642	0.597	0.205	0.698	0.493	0.560	0.607	0.173	0.789	0.392	0.497
GDG	0.458	0.382	0.797	0.422	0.540	0.400	0.299	0.742	0.362	0.481	0.679	0.454	0.698	0.560	0.614	0.719	0.483	0.815	0.559	0.65
PisCES	0.000	0.000	0.464	**1.000**	0.631	0.000	0.000	0.427	**1.000**	0.596	0.149	0.046	0.280	0.907	0.372	0.325	0.082	0.490	0.665	0.440
sE-NMF	0.341	0.187	0.652	0.610	0.628	0.386	0.265	0.641	0.625	0.628	0.670	0.279	0.74	0.406	0.519	0.659	0.269	0.813	0.373	0.504
CCPSO	0.639	0.409	**1.000**	0.392	0.555	0.624	0.351	**1.000**	0.333	0.491	**0.853**	**0.659**	**0.918**	0.695	**0.789**	**0.839**	0.659	**0.959**	0.659	0.772
ComSP (**ours**)	**0.722**	**0.775**	**0.866**	**0.839**	**0.852**	**0.735**	**0.775**	**0.846**	**0.858**	**0.850**	**0.806**	**0.635**	0.673	**0.921**	**0.769**	**0.808**	**0.680**	0.735	**0.898**	**0.797**

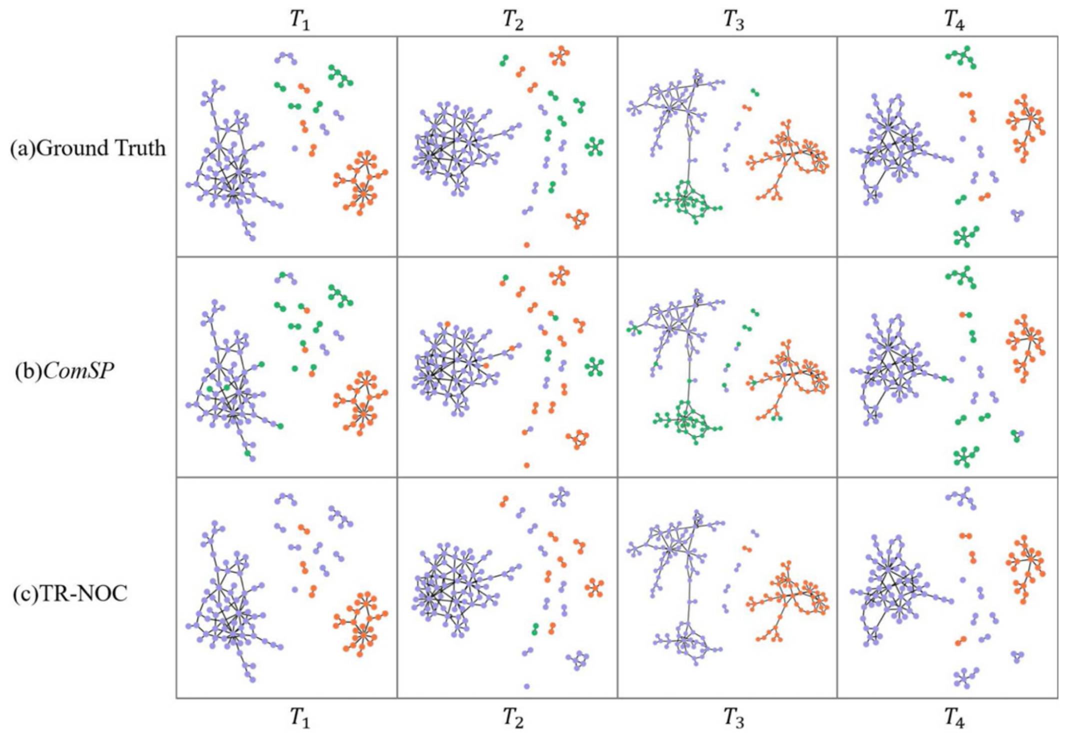


Figure 4. The visualization of community detection results by our approach (ComSP) (b) and TR-NOC (c) on the network sequence of Reddit-I(a), in comparison with the ground truth (c). Different colors in the plots correspond different community labels in the ground truth, respectively. Here, the ratio of correctly clustering on each snapshot of the dynamic network is $[0.917, 0.854, 0.907, 0.932]$ for ComSP and $[0.900, 0.846, 0.787, 0.846]$ for TR-NOC, respectively.

superior to our method in terms of precision, our method has much higher recall values than CCPSO. By looking into the precisions of CCPSO which are close to 1, we find that CCPSO generates a lot of communities in each snapshot, independent of the aligning community operation. As a result, CCPSO performs well in Reddit-III which consists of a large number of small communities.

To get more insights into the differences between the proposed method and its rival TR-NOC, we visualize the partition results of the two methods for each snapshot of Reddit-I(a) with the ground truth as a reference. It becomes evident in figure 4 that the communities discovered by our method accord with the ground truth in tracking the community evolution. Specifically, there are three ground-truth clusters (labeled in three colors) and two major components in the first snapshot which are successfully identified by the proposed method. However, TR-NOC neither unfolds three clusters in the first snapshot, nor tracks the split of the second largest component at the second time step. More noticeably, in the challenging situation of community organization at the third time step where there are some bridging nodes between two communities (as shown in the ground truth), our method provides basically clear separation between these two communities with only one error-tagging node on the boundary. In contrast, TR-NOC mixes two communities (in violet and green, respectively) together. We can conclude from the visualized results that TR-NOC tends to form large communities in essence, which explains why the recall of TR-NOC is remarkably higher than most of the baseline methods(as shown in figure 3).

Furthermore, we compare the performance of our method with TR-NOC in identifying the community affiliations of the top-10% nodes in average clustering coefficient[8] in dynamic networks. In fact, tracking the community affiliations of critical nodes has its own merits in some fields such as neuroscience [56]. In the experiments, we use this task to evaluate the performance of our method. Specifically, we rank the nodes according to their average clustering coefficient in the period of interest, where the average clustering coefficient is defined as the local clustering coefficients of a node in all snapshots averaged over time. Then we select the nodes whose average clustering coefficient is in the top-10% and spot the community memberships of these nodes in Reddit datasets with different time spans. Table 4 shows the superiority of our method on four datasets as a whole. Note that recall of TR-NOC remains higher than ours in this application, as discussed above.

[8] The average clustering coefficient is computed by $\bar{c}(i) = \frac{1}{T} \times \sum_{t=1}^{T} c_t(i)$, where $c_t(i)$ is the clustering coefficient of node i in snapshot G_t.

Table 4. Performance comparisons between ComSP and TR-NOC on the identification of community affiliations of the top-10% nodes in terms of average clustering coefficient in Reddit.

	Reddit-I(a)					Reddit-I(b)					Reddit-III(a)					Reddit-III(b)				
Method	NMI	ARI	P	R	F_1	NMI	ARI	P	R	F_1	NMI	ARI	P	R	F_1	NMI	ARI	P	R	F_1
TR-NOC	0.733	0.590	0.767	**1.000**	0.868	0.000	0.000	0.432	**1.000**	0.603	**0.930**	**0.745**	0.802	**1.000**	**0.890**	0.853	0.636	0.672	**0.963**	0.791
ComSP (**ours**)	**0.854**	**0.870**	**0.938**	0.925	**0.931**	**0.746**	**0.797**	**0.851**	0.831	**0.841**	0.916	0.665	**0.919**	0.805	0.858	**0.892**	**0.709**	**0.879**	0.781	**0.827**

7. Conclusion

In this work, we have proposed a novel dynamic community discovery algorithm, which projects each snapshot into a common subspace to produce a global smoothing for each snapshot, and clusters on all time-stamped nodes in dynamic networks. This way, our method gains the best stability performance in dynamic networks, compared to the state-of-the-art methods. Another advantage of our method is, by clustering the nodes in the projected subspace, that community detection and tracing are performed in one stage. Compared with the two-stage methods, the one-stage method omits the matching stage for the sequential snapshots and reduces the computational complexity. However, we also note that one limitation of our method, like other one-stage approaches, is that our method is off-line, which means we cannot yet detect communities of dynamic networks in real time.

We have evaluated the proposed method on both real and synthetic datasets and demonstrated that it performs more stably than the baselines. The sizes of communities discovered by the proposed method are more close to the GT, that is, the number of communities is almost the same as the ground truth, indicating that our method can successfully trace most of the communities. However, the recall rate of our method is inferior to the state-of-the-art method. We believe this is due to clustering based on the constructed node similarity matrix which tends to neglect some details of temporal connection patterns among nodes. To mitigate the influence of scattered nodes and increase the purity of clusters is an important part for our future work.

Acknowledgments

This work is supported by the National Science Foundation of China (Nos. 61873218 and 61973086), Sichuan Province Science and Technology Department (No. 18YYJC1147), Nanchong school cooperation project (No. 18SXHZ0009), Shanghai Municipal Science and Technology Major project (No. 2018SHZDZX01) and ZJ Lab, and SWPU Innovation Base Foundation (No. 642).

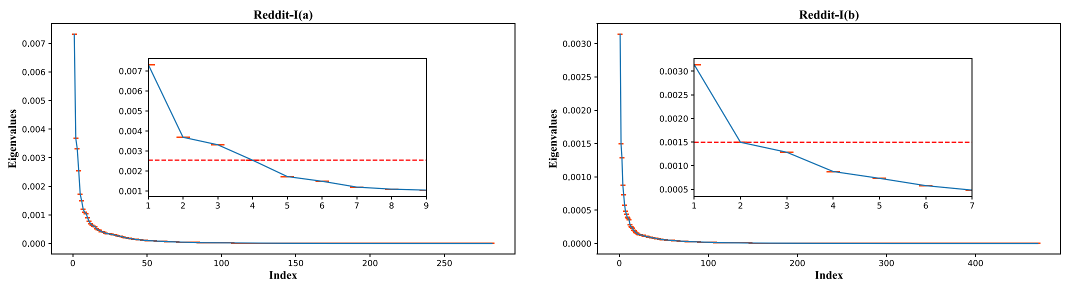
Data availability

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

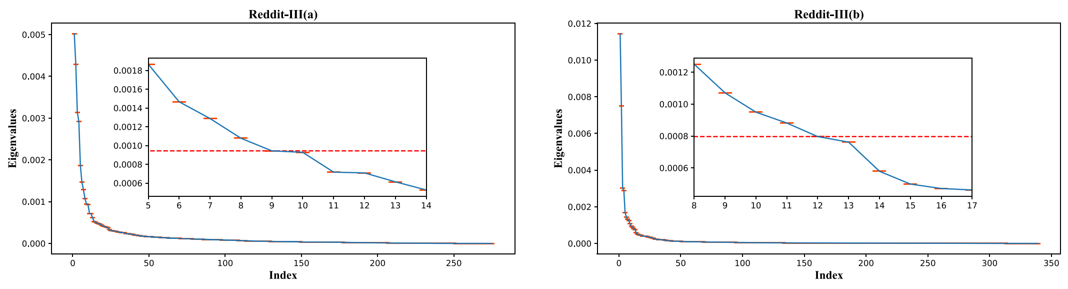
Appendix A. Selection of parameter *s* and *K*

In our algorithm, the optimal subspace is composed of principal eigenvectors and can render the clustering pattern of nodes, based on similarity matrix. In the light of previous work [57] where it has been shown that for a network with N nodes and m communities, there will typically be m eigenvalues that are particularly larger than the magnitudes of all the other $(N - m)$ eigenvalues. That is, there are significant gaps between m eigenvalues and the remainder. Thus we choose the value of s in a similar way: we select s leading eigenvalues that distance themselves from the majority of eigenvalues with low values, based on the eigenvalue distribution of the covariance matrix (as shown in figure 5).

Regarding the number of communities, we also heuristically approximate it according to the aforementioned observations. As is shown in figure 5, the prominent eigen-gaps indicate the existence of clustering pattern when nodes are embedded in the eigen-subspace, and the clusters in the summarized similarity relationships given by M most likely correspond to the communities present in the dynamic network. In particular, the optimal number of communities is expected to be close to the value of s, which makes it convenient for K-means to search the appropriate number of clustering. Therefore, we use the elbow method to search the optimal number of clusters around the value of s. For example, for Reddit-I(a) we first determine the optimal number of eigenvalues, i.e. $s = 4$, then we search the cluster number around 4, as shown in figure 6, where the metrics is the sum of squared distances to the closest centroid for all observations in K-means. It can be found that the optimal K is on the ‘elbow’ of the curve, where $K = 3$.



(a) (b)



(c) (d)

Figure 5. The distributions of the eigenvalues for similarity matrix, where the inset plots show significant eigen-gaps between leading eigenvalues.

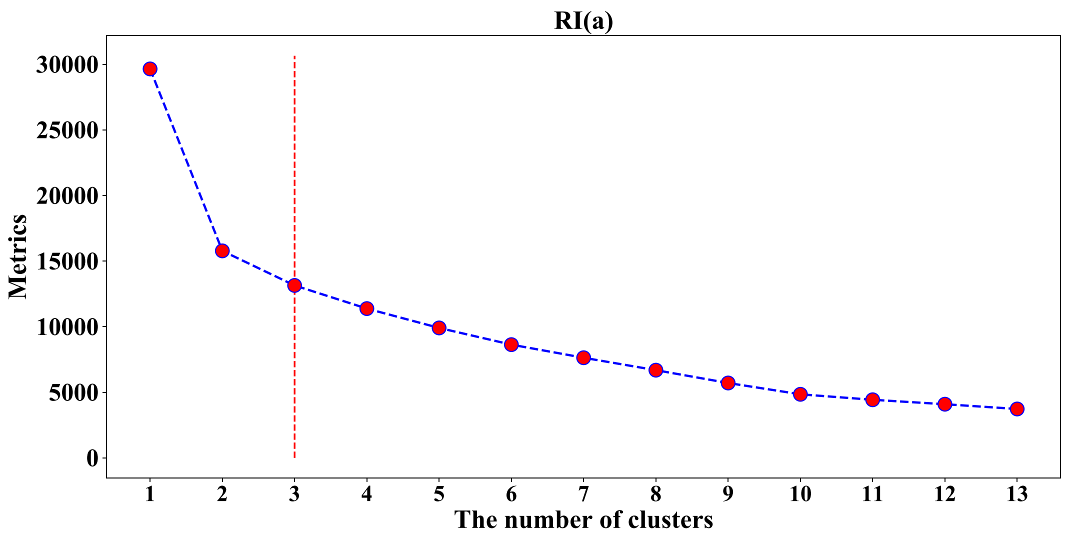


Figure 6. The elbow method to search K in Reddit-I(a), where we select $K = 3$ (dashed red line) in our experiment.

Appendix B. Evaluation metrics

Let $|S|$ be the number of nodes, $L(i)$ be the real community of node i, and $|L(i)|$ be the size of the community $L(i)$. $\psi(i,j)$ is 1 if and only if nodes i and j are correctly detected in the same community, then Bcubed measure can be written as follows:

$$F_1 = \frac{2 \times P \times R}{P + R}, \tag{B.1}$$

where

$$P = \frac{1}{|S|}\sum_{i=1}^{|S|}\left(\frac{1}{|C(i)|}\sum_{j\in C(i)}\psi(i,j)\right), \tag{B.2}$$

and

$$R = \frac{1}{|S|}\sum_{i=1}^{|S|}\left(\frac{1}{|L(i)|}\sum_{j\in L(i)}\psi(i,j)\right). \tag{B.3}$$

We draw a statistical matrix N with the dimensions $|C| \times |L|$, where $|C|$ is the number of detected communities. N_{cl} is the number of nodes which belong to the cth predicted community and the lth true community at the same time, then $N_{c\cdot}$ and $N_{\cdot l}$ are the number of nodes in the cth predicted community and the lth true community, respectively. Therefore, NMI is given by:

$$\mathrm{NMI} = \frac{\mathrm{MI}(C,L)}{\sqrt{H(C)H(L)}}, \tag{B.4}$$

where

$$\mathrm{MI}(C,L) = \sum_{c=1}^{|C|}\sum_{l=1}^{|L|}\frac{N_{cl}}{|S|}\log(|S|\frac{N_{cl}}{N_{c\cdot}N_{\cdot l}}), \tag{B.5}$$

$$H(C) = -\sum_{c=1}^{|C|}\frac{N_{c\cdot}}{|S|}\log(\frac{N_{c\cdot}}{|S|}), \tag{B.6}$$

and

$$H(L) = -\sum_{l=1}^{|L|}\frac{N_{\cdot l}}{|S|}\log(\frac{N_{\cdot l}}{|S|}). \tag{B.7}$$

Besides, the corrected-for-chance version of the rand index called adjusted rand index [52] is employed with a statistical matrix N, which reads as:

$$\mathrm{ARI} = \frac{\mathrm{RI} - E(\mathrm{RI})}{\mathrm{Max}(\mathrm{RI}) - E(\mathrm{RI})}, \tag{B.8}$$

where RI is the rand index [58]:

$$\mathrm{RI} = \sum_{c,l}\binom{N_{cl}}{2}, \tag{B.9}$$

and, Max(RI) and E(RI) are the maximum index and the expected index respectively:

$$\mathrm{Max}(\mathrm{RI}) = \frac{1}{2} \times \left[\sum_{l}\binom{N_{\cdot l}}{2} + \sum_{c}\binom{N_{c\cdot}}{2}\right], \tag{B.10}$$

$$E(\mathrm{RI}) = \frac{\sum_{l}\binom{N_{\cdot l}}{2} \times \sum_{c}\binom{N_{c\cdot}}{2}}{\binom{|S|}{2}}. \tag{B.11}$$

Note that, $\binom{|S|}{2}$ is the number of all possible node pairs

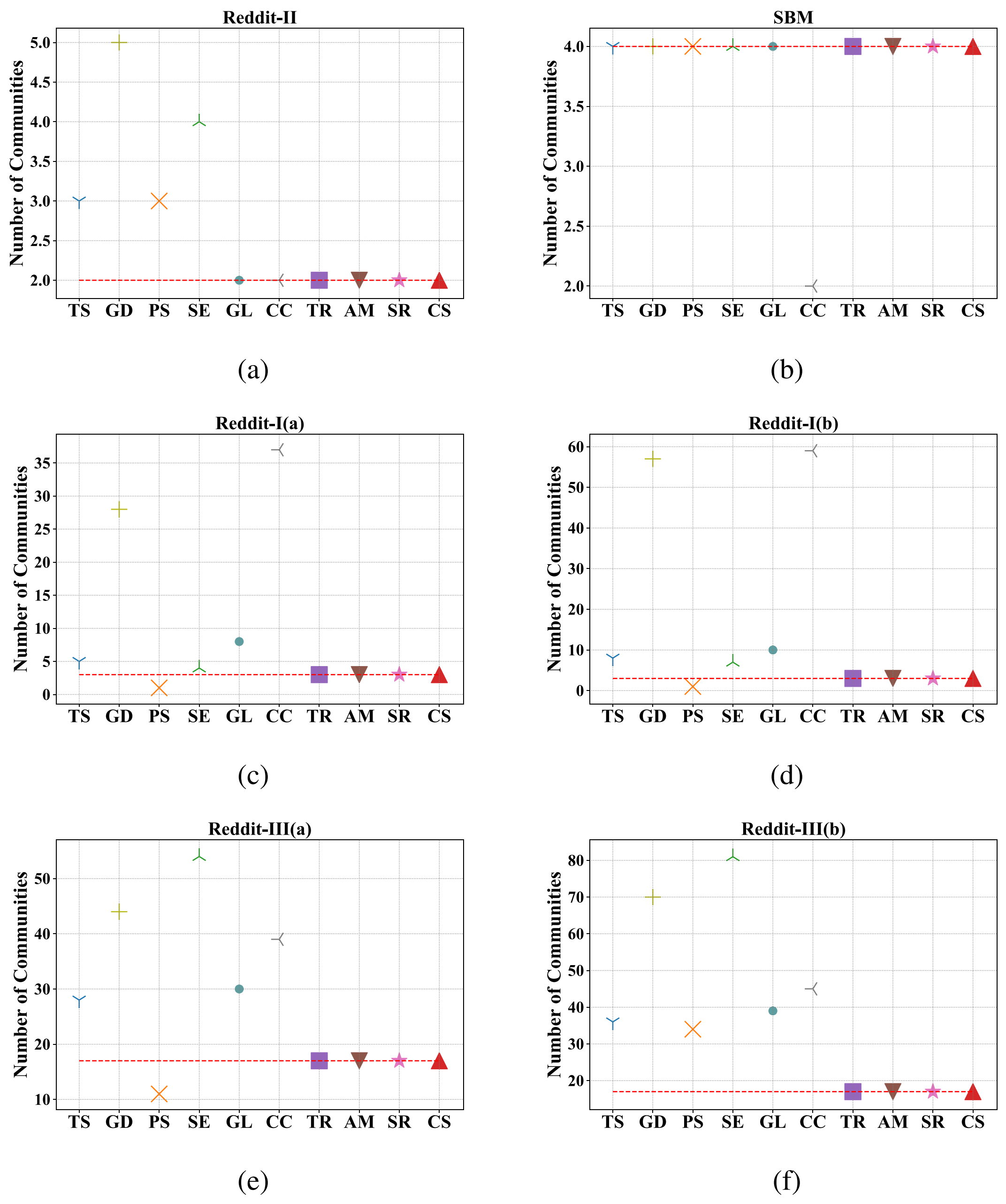


Figure 7. Number of community partitions on SBM (subplot (a)), Reddit-II (subplot (b)) Reddit-I (subplots (c)–(d)) and Reddit-III(subplots (e)–(f)), where each point corresponds to the number of communities detected by eight methods, i.e. TR-NOC (TR), PisCES (PS), ts-Spect (TS), GDG (GD), sE-NMF (SE), CCPSO (CC), MultiGL(GL), Adj-Mat (AM), SuRep (SR) and ComSP (CS), respectively.

Appendix C. Complementary visualization of experimental results

We visualize some results here complementary to the tables and figures in the main text. The number of communities detected by each method is shown in figure 7. Moreover, we transform table 2 into the figure to visualize the results of experiments, as shown in figure 8. However, for the methods whose scores are very close or equal to zero, it is not direct to recognize them via figure.

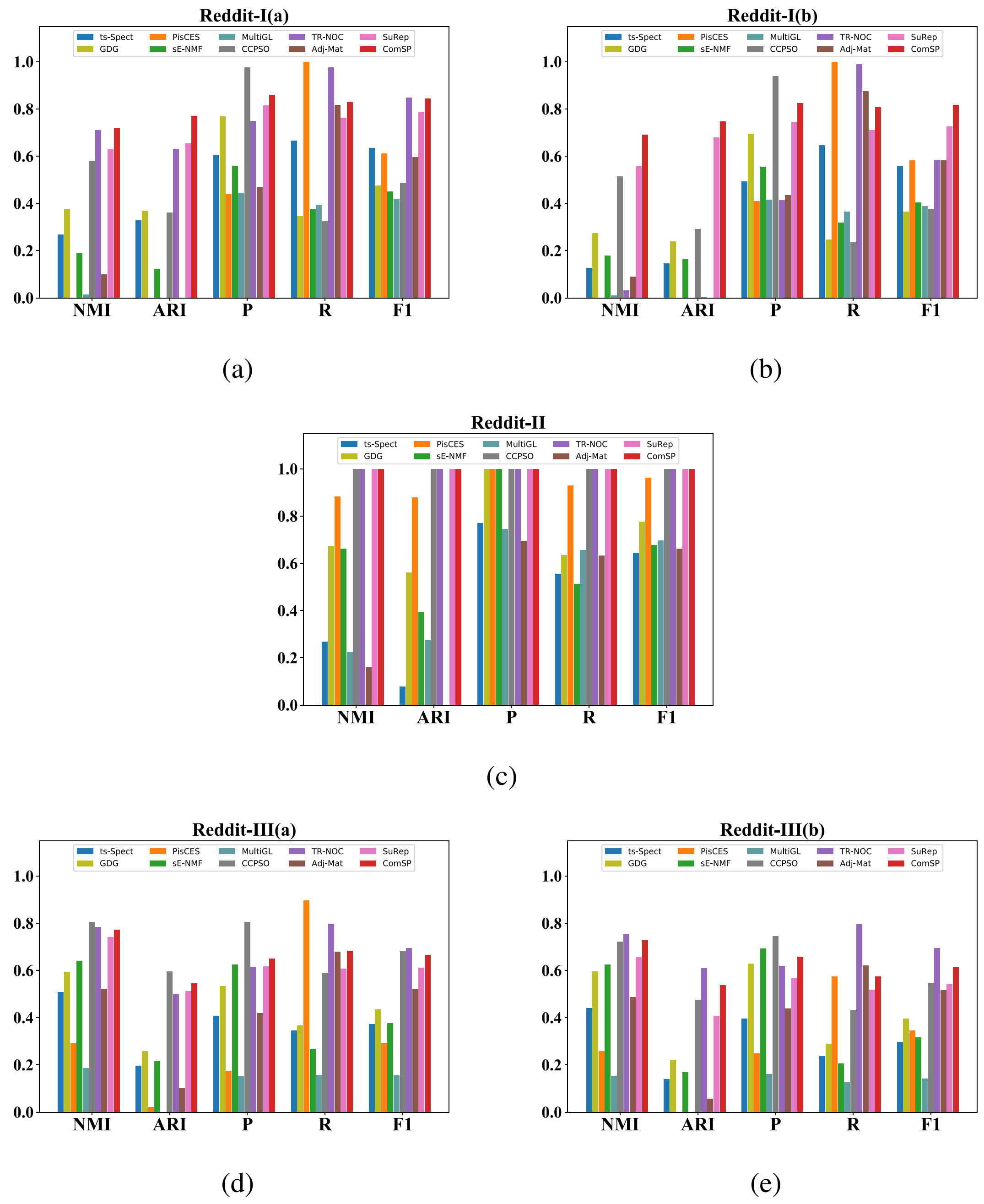


Figure 8. The evaluation of the methods on the real datasets, i.e. TR-NOC (TR), PisCES (PS), ts-Spect (TS), GDG (GD), sE-NMF (SE), CCPSO (CC), MultiGL(GL), Adj-Mat (AM), SuRep (SR) and ComSP (CS).

References

[1] Aynaud T and Guillaume J L 2010 Static community detection algorithms for evolving networks *8th Int. Symp. on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks* (IEEE) pp 513–9

[2] Li P, Chen K, Ge Y, Zhang K and Small M 2018 *Europhys. Lett.* **120** 28003

[3] Bonchi F, Castillo C, Gionis A and Jaimes A 2011 *ACM Trans. Intell. Syst. Technol.* **2** 22

[4] Serrat O 2017 *Social Network Analysis* (Berlin: Springer)

[5] Du N, Wu B, Pei X, Wang B and Xu L 2007 Community detection in large-scale social networks *Proc. of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis* pp 16–25

[6] Haggerty L S *et al* 2013 *Mol. Biol. Evol.* **31** 501–16

[7] Li Y, He K, Kloster K, Bindel D and Hopcroft J 2018 *ACM Trans. Knowl. Discovery Data* **12** 17

[8] Tian B and Wang L 2017 *Neurocomputing* **253** 34–41

[9] Ozer M, Kim N and Davulcu H 2016 Community detection in political twitter networks using nonnegative matrix factorization methods *Proc. of the 2016 IEEE/ACM Int. Conf. on Advances in Social Networks Analysis and Mining* (IEEE) pp 81–8

[10] Cortes C, Pregibon D and Volinsky C 2001 *Communities of Interest Int. Symp. on Intelligent Data Analysis* (Berlin: Springer) pp 105–14

[11] Khan B S and Niazi M A 2017 arXiv:1708.00977
[12] Fortunato S 2010 *Phys. Rep.* **486** 75–174
[13] Newman M E J 2001 *Proc. Natl Acad. Sci.* **98** 404–9
[14] Shang J, Liu L, Xie F, Chen Z, Miao J, Fang X and Wu C 2014 arXiv:1407.2683
[15] Chen W, Liu Z, Sun X and Wang Y 2010 *Data Min. Knowl. Disc* **21** 224–40
[16] Zhang L, Takahashi D, Hartvig M and Andersen K H 2017 *Proc. R. Soc. B.* **284** 1772
[17] Hopcroft J, Khan O, Kulis B and Selman B 2003 Natural communities in large linked networks *Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining* (ACM) pp 541–6
[18] Cazabet R and Rossetti G 2019 Challenges in community discovery on temporal networks *Temporal Network Theory* (Berlin: Springer) pp 181–97
[19] Aynaud T, Fleury E, Guillaume J L and Wang Q 2013 *Communities in Evolving Networks: Definitions, Detection, and Analysis Techniques* (Berlin: Springer) pp 159–200
[20] Lin Y R, Chi Y, Zhu S, Sundaram H and Tseng B L 2009 *ACM Trans. Knowl. Discovery Data* **3** 8
[21] Chen K, Yu L, Zhu T, Li P and Kurths J 2019 *IEEE Trans. Knowl. Data Eng.* 1–10
[22] Newman M E J 2012 *Nat. Phys.* **8** 25–31
[23] Newman M E J 2004 *Eur. Phys. J.* B **38** 321–30
[24] Ye Z, Hu S and Yu J 2008 *Phys. Rev.* E **78** 046115
[25] Gao C *et al* 2018 *Ann. Stat.* **46** 2153–85
[26] Dhumal A T, Narayanan R G and Kumar G S 2012 *IJMIC* **15** 164–72
[27] Sobolevsky S, Campari R, Belyi A and Ratti C 2014 General optimization technique for high-quality community detection in complex networks *Phys. Rev.* E **90** 012811
[28] Boccaletti S, Ivanchenko M, Latora V, Pluchino A and Rapisarda A 2007 *Phys. Rev.* E **75** 045102
[29] Von Luxburg U 2007 *Stat. Comput.* **17** 395–416
[30] Mahmood A, Small M, Al-Maadeed S A and Rajpoot N 2017 *IEEE Trans. Knowl. Data Eng.* **29** 921–35
[31] He K, Shi P, Bindel D and Hopcroft J 2019 *ACM Trans. Knowl. Discovery Data* **13** 52
[32] Rossetti G and Cazabet R 2018 *ACM Comput. Surv.* **51** 35
[33] Gauvin L, Panisson A and Cattuto C 2014 *PloS one* **9** e86028
[34] Sarantopoulos I, Papatheodorou D, Vogiatzis D, Tzortzis G and Paliouras G 2018 TimeRank: a random walk approach for community discovery in dynamic networks *Int. Conf. on Complex Networks and Their Applications* (Berlin: Springer) pp 338–50
[35] Liu F, Choi D, Xie L and Roeder K 2018 *Proc. Natl Acad. Sci. USA* **115** 927–32
[36] Ma X and Dong D 2017 *IEEE Trans. Knowl. Data Eng.* **29** 1045–58
[37] Zeng X, Wang W, Chen C and Yen G G 2019 *IEEE Trans. Cybern.* 1–12
[38] Zhang Y Q, Li X, Xu J and Vasilakos A V 2014 *IEEE Trans. Syst. Man Cybern.* **45** 214–22
[39] Viard T, Latapy M and Magnien C 2016 *Theor. Comput. Sci.* **609** 245–52
[40] Wang W and Li X 2019 *IEEE Trans. Netw. Sci. Eng.* **7** 1508–20
[41] Mucha P J, Richardson T, Macon K, Porter M A and Onnela J-P 2010 *Science* **328** 876–8
[42] Arthur D and Vassilvitskii S 2007 *k*-means++: the advantages of careful seeding *Proc. of the 18th annual ACM-SIAM Symp. on Discrete algorithms* (SIAM) 1027–35
[43] Wang Y J and Wong G Y 1987 *J. Am. Stat. Assoc.* **82** 8–19
[44] Granell C, Darst R K, Arenas A, Fortunato S and Gómez S 2015 *Phys. Rev.* E **92** 012805
[45] Holland P W, Laskey K B and Leinhardt S 1983 *Soc. Netw.* **5** 109–37
[46] Tarrés-Deulofeu M, Godoy-Lorite A, Guimerà R and Sales-Pardo M 2019 *Phys. Rev.* E **99** 032307
[47] Guimerà R and Sales-Pardo M 2009 *Proc. Natl Acad. Sci.* **106** 22073–8
[48] Cazabet R and Amblard F 2014 *Encyclopedia of Social Network Analysis and Mining* (Berlin: Springer) pp 404–14
[49] Lucas G S J, Marya B, Inderjit S J and Peter J M 2011–2019 https://github.com/GenLouvain/GenLouvain(2011-2019)
[50] Wu Z, Cao J, Zhu G, Yin W, Cuzzocrea A and Jin S 2015 *World Wide Web* p 18
[51] Strehl A and Ghosh J 2002 *J. Mach. Learn. Res.* **3** 583–617
[52] Hubert L and Arabie P 1985 *J. Classif.* **2** 193–218
[53] Amigó E, Gonzalo J, Artiles J and Verdejo F 2009 *Inf. Retr.* **12** 461–86
[54] Gong M, Cai Q, Li Y and Ma J 2012 An improved memetic algorithm for community detection in complex networks *2012 IEEE Congress on Evolutionary Computation* (Piscataway, NJ: IEEE) pp 1–8
[55] Newman M E J 2006 *Proc. Natl Acad. Sci.* **103** 8577–82
[56] Bassett D S, Wymbs N F, Porter M A, Mucha P J, Carlson J M and Grafton S T 2011 *Proc. Natl Acad. Sci.* **108** 7641–6
[57] Chauhan S, Girvan M and Ott E 2009 *Phys. Rev.* E **80** 056114
[58] Rand W M 1971 *J. Am. Stat. Assoc.* **66** 846–50