Social Network Analysis and Mining (2021) 11:67
https://doi.org/10.1007/s13278-021-00774-8

ORIGINAL ARTICLE

Summable and nonsummable data-driven models for community detection in feature-rich networks

Soroosh Shalileh[1,2] · **Boris Mirkin**[1,3]

Received: 25 January 2021 / Revised: 10 July 2021 / Accepted: 13 July 2021 / Published online: 28 July 2021

Abstract
A feature-rich network is a network whose nodes are characterized by categorical or quantitative features. We propose a data-driven model for finding a partition of the nodes to approximate both the network link data and the feature data. The model involves summary quantitative characteristics of both network links and features. We distinguish between two modes of using the network link data. One mode postulates that the link values are comparable and summable across the network (summability); the other assumption models the case in which different nodes represent different measurement systems so that the link data are neither comparable, nor summable, across different nodes (nonsummability). We derive a Pythagorean decomposition of the combined data scatter involving our data recovery least-squares criterion. We address an equivalent problem of maximizing its complementary part, the contribution of a found partition to the combined data scatter. We follow a doubly greedy strategy in maximizing that. First, communities are found one-by-one, and second, entities are added one-by-one in the process of identifying a community. Our algorithms determine the number of clusters automatically. The nonsummability version proves to have a niche of its own; also, it is faster than the other version. In our experiments, they appear to be competitive over generated synthetic data sets and six real-world data sets from the literature.

Keywords Attributed network · Feature-rich network · Community detection · Sequential extraction · Least squares data recovery · One-by-one clustering

1 Introduction: background, motivation, and previous work

1.1 Background

Community detection in networks is a popular topic applied in various domains ranging from sociology to biology to computer science. A network is a set of objects, usually referred to as nodes, which are interconnected by pair-wise links. Typical examples are: a network of mutual friendship relations between a set of individuals; a network of enterprises related by mutual supplies; and a network of websites related by mutual visits. When nodes are additionally supplied with a set of features characterizing them, such a network is referred to as a feature-rich (Interdonato et al. 2019), or node-attributed (Bojchevski and Günnemann 2018; Xu et al. 2012), network. In a friendship network, features may characterize individual's demographics, background, interests, etc.

A community is a group of relatively densely inter-connected nodes that are similar in the feature space too.

Figure 1a illustrates the concept of feature-rich network as a formal structure, and Fig. 1b visualizes communities in a network.

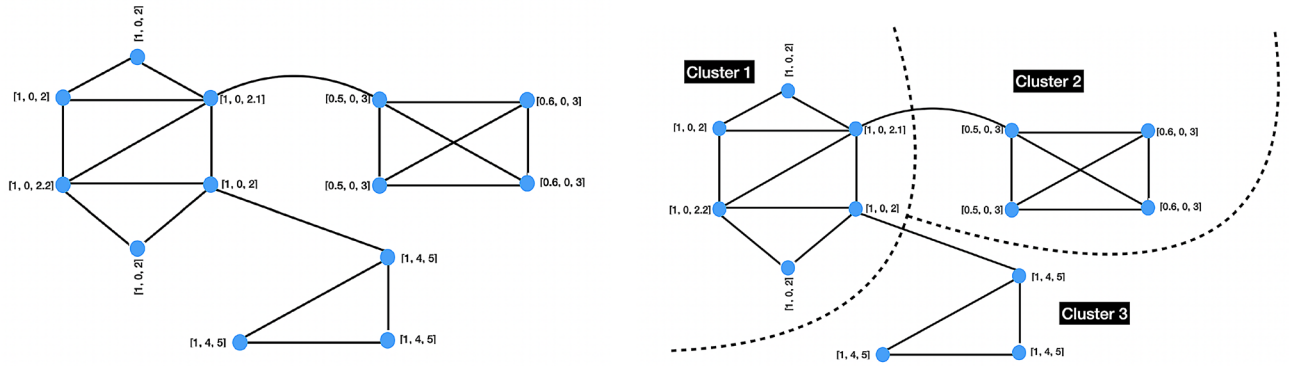
There have been published a number of papers proposing various approaches to identifying communities in feature-rich networks. A comprehensive, yet concise, review of methods for community detection in feature-rich networks can be found in Chunaev (2020). In this review, all the community detection methods are classified according to the

✉ Boris Mirkin
bmirkin@hse.ru

[1] Department of Data Analysis and Artificial Intelligence, HSE University, Pokrovsky Boulevard, 11, Moscow, Russian Federation

[2] Laboratory of Methods for Big Data Analysis, HSE University, Pokrovsky Boulevard, 11, Moscow, Russian Federation

[3] Department of Computer Science and Information Systems, Birkbeck University of London, Malet Street, London WC1E 7HX, UK



(a) Feature-rich network (b) Detected clusters

Fig. 1 Communities in a feature-rich network: **a** visualizes the data structure, **b** presents a network whose nodes are partitioned in communities

stage of the process of finding communities at which the two data sources, network and features, are merged together. The merger may occur before the process begins (early fusion), within the process (simultaneous fusion), and after the process (late fusion).

1.2 Motivation

The subject of our interest belongs to the simultaneous fusion stage, at which both data sources, network links and feature values, are available for investigation. We are going to develop a mathematical model for the data, so that the model is able to help us in detecting communities in the network informed by the data. Among mathematical data modeling approaches, we distinguish between theory-driven and data-driven approaches. Theory-driven approaches involve a model of the world leading to a probabilistic distribution, parameters of which can be recovered from the data. In contrast, data-driven approaches involve no world models but rather focus on modeling the data as is.

Our data-driven model conventionally assumes that there is a hidden partition of the node set in nonoverlapping communities, which is supplied with hidden parameters encoding the average link intensities in the network and community central points, in the feature space. These are used at the "decoding" stage so that the residuals of data modeling equations are minimized according to the least squares criterion. Such an approach is referred to as the data recovery approach in Mirkin (2012); in neural network domain, that is referred to as auto-encoder (Ng 2011). Unsupervised data analysis methods such as K-means clustering and principal component analysis naturally fall within this approach, as described in Mirkin (2012).

As usual, the least squares criterion leads to computationally hard problems which are tackled with various heuristics. In particular, we follow a strategy of sequentially extracting clusters one-by-one. This strategy naturally fits into the additive structure of the least squares criterion. Previously, this strategy has been applied separately to only network, or entity-to-entity similarity data, and to only feature data. Applied to similarity data, it was first described in English in Mirkin (1987) and experimentally validated in papers like (Mirkin and Nascimento 2012). Similar constructions, for dissimilarity data, have been developed in Vichi (2008). Applied to feature space data, the strategy was experimentally validated in Chiang and Mirkin (2010), Amorim and Mirkin (2012).

Recently, we applied this approach to network data and feature space data combined (Shalileh and Mirkin 2020). This paper significantly extends that to a more comprehensive analysis of the network structure.

First of all, we introduce and test here two different modes of using network link structure. Specifically, we now distinguish between summable and nonsummable modes. The former corresponds to the case at which all link weights are measured in the same scale, so that they are comparable and summable across the entire data table. In the nonsummable mode, each node's links are considered as measured in different scales.

This assumption points to a not uncommon data type emerging, for example, in some psychological experiments in which the entities are individuals or cognitive subsystems with different scales of individual judgments. Whenever the node's links are measured independently of the other nodes, there is a potential for the weights to be nonsummable. To give an example of nonsummable links, let us consider two sets of internet sites: sites in one set provide classical music education, and sites in another set sell goods. These two sets would usually much differ in at least two aspects. First, the numbers of visitors are of different orders at these sets: the numbers are massive at selling goods sites; in contrast, the numbers of visitors

at classical music education would be smaller by several orders of magnitude. Second, the time spent, in general, would be much different at these two sets: seconds at purchasing goods and hours at listening music. As we will see, taking into account the nonsummability phenomenon, even in the context of ordinary feature-rich networks, leads to advantages in at least two aspects: the speed of computation and quality of cluster recovery at some data types.

We apply the least squares approach to both cases, leading to two different versions of the method, and conduct a comprehensive set of experiments to validate and to compare the performance of the newly proposed algorithms. Second, we expand here both the list of real-world data sets and the list of algorithms under comparison. Specifically, we add to our collection of small-sized data sets a medium-sized data set with 3490 nodes, which may add substance to our claim that our method works well for both types of data sets. Also, we found a recent heuristic algorithm, EVA (Citraro and Rossetti 2020), which has a publicly available code, so that we were able to add that to the list of our competition. We inserted a section discussing the complexity of our algorithms and provided a comparison of the timing taken by all the algorithms under consideration. It appears the algorithm in the nonsummable mode works almost as fast as the fastest out of our sample.

It is noteworthy to add that our method:

- Is data driven;
- Admits either quantitative or categorical features or both;
- Involves an explicit relative weighting of the two data sources in the fitting criterion;
- Assumes that hidden communities are crisp and nonoverlapping;
- Determines the number of communities automatically.

Our method finds communities one-by-one, which leads to a natural way for selecting the number of clusters. All procedures involved are finite and, thus, always convergent. Our experiments show that this approach is able to recover hidden clusters in feature-rich networks reasonably well. Moreover, it is competitive against existing state-of-the-art approaches.

The rest of the paper is organized as follows. Section 1.3 reviews the previous work. We describe our models and algorithms in Sect. 2. Section 3 is devoted to setting of our experiments for validation of the algorithms and comparison of them with some state-of-the-arts algorithms. It presents: (a) competition; (b) data sets, both real-world and artificially generated; (c) criteria for assessment of the quality of experiments. In Sects. 4 and 5 we describe results of our experiments. We draw conclusions in Sect. 6.

1.3 Previous work

Within the simultaneous fusion approach (Chunaev 2020) literature, we consider three directions: (a) heuristics, (b) theory-driven modeling, and (c) data-driven modeling approaches.

We are going to briefly discuss these three in the remainder of this section after a mention of some classical clustering methods.

These classical methods include the normalized cut and related spectral clustering (Shi and Malik 2000), as well as the modularity-based method (Newman 2006; Dang and Viennet 2010). The Louvain algorithm (Blondel et al. 2008) detects communities by locally maximizing the modularity score. The most recent reviews of research on community detection in networks with no feature data include a comprehensive monograph (Doreian et al. 2020) as well as thought-provoking reviews (Javed et al. 2018; Hoffman et al. 2018). A review of the theory-driven approach in the analysis of networks (Goldenberg et al. 2010) should be mentioned too.

Among heuristics approaches, one can distinguish those at which criteria of the classical clustering algorithms are modified according to the presence of two data sources. Paper (Ye et al. 2017) modifies the normalized cut criterion by adding the so-called unimodality compactness to reflect the homogeneity of attributes within the community. A modified modularity criterion and corresponding method is developed in Sánchez et al. (2015). A modified Louvain method is proposed and tested in Combe et al. (2015). Another popular direction of development in this area is the so-called network embedding (see (Chang et al. 2015; Cavallari et al. 2017; Sun et al. 2020)). In these approaches, both the network and feature data are approximated with a low-dimension Euclidean vector space.

The theory-driven approach involves both the maximum likelihood and Bayesian criteria for fitting probabilistic models. Many methods in this category involve stochastic block models (SBM) which have been successfully used for detection of communities in conventional networks. In Stanley et al. (2019) network structures are modeled with SBM, while the continuous features are modeled with a Gaussian mixture model. The Blockmodel Entropy Significance Test (BESTest) (Peel et al. 2017) for evaluation of how much a metadata partition is relevant to the network structure. The BESTest works by first dividing network's nodes according to the feature labels and then by computing the entropy of that SBM which best corresponds to the partition.

Methods in Xu et al. (2012), Newman and Clauset (2016), Bojchevski and Günnemann (2018) are based on Bayesian inferences. In Yang et al. (2013) the authors proposed clustering criterion to statistically model interrelation between the network structure and node attributes.

As to the data-driven modeling approach, it seems research in this direction is rather scarce. Some authors propose the so-called non-negative matrix factorization (NNMF), which is a technique to approximate the data via data matrix factorization into non-negative matrices of simpler structure. In papers (Wang et al. 2016; Cao et al. 2019) combined criteria for such an approximation and methods for suboptimally solving them are proposed. The criteria are based on the least-squares approach. However, in contrast to our line of thinking, these criteria involve some derived data rather than the original ones. A different approach is described in Akoglu et al. (2012). Here, the data are summarized as given; the quality, however, is scored according to the principle of minimum description length (MDL) so that the number of bits in coding of the summary is minimized.

One may say that our approach combines aspects of the two approaches above: a straightforward modeling of the data as is, like in Akoglu et al. (2012), and a least-squares criterion, like in Wang et al. (2016), Cao et al. (2019).

2 Methodology

2.1 Data recovery model for community detection

Consider a network with features at the nodes, $A = \{P, Y\}$, over an entity set I. Here I is a set of network nodes of cardinality $|I| = N$; $P = (p_{ij})$ is an $N \times N$ matrix of mutual link weights between nodes $i, j \in I$; and $Y = (y_{iv})$ is an $N \times V$ matrix of feature values, so that entry y_{iv} is the value of feature $v = 1, 2, \ldots, V$ at node $i \in I$. This definition covers a wide range of networks, including, for example, a flat network in which inter-node links simply exist or not, but have no associated weights. Such a network can be represented by matrix P at which $p_{ij} = 1$ if a link between i and j exists, and $p_{ij} = 0$ if not.

To build a data-driven community model, let us specify the following notation.

A community, or cluster, $S \subset I$ is represented by a binary $N \times 1$ membership column vector, $s = (s_i)$ in which $s_i = 1$ if $i \in S$, and $s_i = 0$, otherwise ($i = 1, 2, \ldots, N$).

In the feature space, community S can be represented by a V-dimensional point $c = (c_v)$, which is a standard to which all the community members relate.

At the network link data, there may be at least two possible assumptions:

(a) AS: Summable weights

This assumption means that the weights p_{ij} are comparable and summable across all the matrix P. In this case, there should be a single intensity weight λ to relate the weights measurement scale to S. Specifically, each within-community weight p_{ij}, $i, j \in S$, in this case, should be large and approximately equal to the intensity λ. The between-community links, ideally, should be all zero.

(b) AN: Nonsummable weights

Under this assumption, weights p_{ij} in any column j are considered incomparable to weights $p_{ij'}$ in any different column $j' \neq j, i, j \in I$. Therefore, at each column $j \in I$ a specific intensity weight λ_j is assumed, so that, for any $i \in S$ the link weights p_{ij} tend to be equal to λ_j.

Extending these definitions to a partition, $S = \{S_1, S_2, \ldots, S_K\}$, of I in K nonoverlapping parts/communities, S can be represented by a binary matrix $s = (s_{ik})$ so that $s_{ik} = 1$ if $i \in S_k$, and $s_{ik} = 0$, otherwise.

To relate any partition to the feature data, we assume that a standard point $c_k = (c_{kv})$ is specified for each community S_k, $k = 1, 2, \ldots, K$, so that approximate equations hold:

$$y_{iv} = \sum_{k=1}^{K} c_{kv} s_{ik} + f_{iv}, i \in I, v \in V. \quad (1)$$

Since communities S_k do not overlap, the sum in the equations plays a rather nominal role: for any $i \in I$, y_{iv} is equal to $c_{kv} + f_{iv}$ just for that k at which $i \in S_k$. The value f_{iv} expresses the extent of approximation and should be made as small as possible.

To approximate the network part of the data, we assume either a total intensity weight λ_k for community S_k, under the summability assumption AS, or column-dependent intensity weights λ_{kj}, under the nonsummability assumption AN ($k = 1, 2, \ldots K; j \in I$). Then the following equations should hold:

$$p_{ij} = \sum_{k=1}^{K} \lambda_k s_{ik} s_{jk} + e_{ij}, i, j \in I, \quad (2)$$

at the AS, and

$$p_{ij} = \sum_{k=1}^{K} \lambda_{kj} s_{ik} + e_{ij}, i, j \in I. \quad (3)$$

at the AN.

Similarly, the sums in these equations are purely nominal. At the AS, they just express that $p_{ij} = \lambda_k$ for all $i, j \in S_k$ ($k = 1, 2, \ldots, K$) or $p_{ij} = 0$, otherwise, up to the residual e_{ij}, of course. At the AN, $p_{ij} = \lambda_{kj}$ for $i \in S_k$ and any $j \in I$, up to small residual e_{ij} again. One may consider that at the AN assumption, the columns $j \in I$ play roles of features.

By using the least-squares approach, we formulate the problem of finding a hidden membership matrix $s = (s_{ik})$, community centers c_k, and intensity weights λ_k or λ_{kj}, as of minimizing the sum of squared residuals:

- at AS assumption:

$$F_{\mathrm{AS}}(\lambda_k, s_k, c_k) = \rho \sum_{k=1}^{K} \sum_{i,v} (y_{iv} - c_{kv} s_{ik})^2 + \xi \sum_{k=1}^{K} \sum_{i,j} (p_{ij} - \lambda_k s_{ik} s_{jk})^2, \tag{4}$$

- at AN assumption:

$$F_{\mathrm{AN}}(\lambda_k, s_k, c_k) = \rho \sum_{k=1}^{K} \sum_{i,v} (y_{iv} - c_{kv} s_{ik})^2 + \xi \sum_{k=1}^{K} \sum_{i,j} (p_{ij} - \lambda_{kj} s_{ik})^2. \tag{5}$$

The factors ρ and ξ in Eqs. (4) and (5) are expert-driven constants to balance the relative weights of the two sources of data, network links and feature values.

Since vectors $s_k = (s_{ik})$ $(k = 1, 2, \ldots, K)$ correspond to a partition, they are mutually orthogonal. That means that for any specific i, s_{ik} is zero for all k's except one: that one k for which S_k contains i. As a result, each of the sums over k in the models relates to a single summand, meaning that the operation of summation over k may be applied outside of the parentheses in Eqs. (4) and (5).

2.2 The iterative extraction approach

The problems of optimization of criteria (4) and (5) are computationally intensive and cannot be solved exactly in a reasonable time. Therefore, there can be various heuristic strategies explored to locally or approximately advance to solving them. We are going to exploit a doubly greedy approach of sequential extraction (Mirkin 2008). This approach can be applied here because the criteria to optimize are additive. According to this approach, parts S_k of the partition S are sought not simultaneously but one-by-one, sequentially, in a greedy manner. That is, a subset of I to serve as S_k at $k = 1$ is found to minimize the part of the criterion related to S_1. Specifically, for an individual community denoted by $T \subseteq I$, its membership by $t = (t_i)$, so that $t_i = 1$ if $i \in T$ and $t_i = 0$, otherwise; its center in feature space, by c; and the corresponding intensity weight by λ (the index k has been removed), the extent of fit between the community and the data set, according to criteria (4) and (5), is

$$f_{\mathrm{AS}}(\lambda, c_v, t_i) = \rho \sum_{i,v} (y_{iv} - c_v t_i)^2 + \xi \sum_{i,j} (p_{ij} - \lambda t_i t_j)^2 \tag{6}$$

at the assumption AS, or

$$f_{\mathrm{AN}}(\lambda_j, c_v, t_i) = \rho \sum_{i,v} (y_{iv} - c_v t_i)^2 + \xi \sum_{i,j} (p_{ij} - \lambda_j t_i)^2 \tag{7}$$

at the assumption AN.

A T locally or approximately minimizing corresponding criterion (6) or (7) is taken as the first part of partition S, S_1. This S_1 is removed from I, and the next part, S_2, is sought in the same way over the residual entity set $I \leftarrow I - S_1$. This continues till a pre-specified stopping criterion is reached, such as, say, when the residual I gets empty.

Given the data matrices, consider a method, *Ext(D)*, for extracting a subset $T \subseteq D$ from any $D \subseteq I$, together with some related quantitative characteristics α, so that $(T, \alpha) = Ext(D)$. Of course, P and Y remain the only data sources used in *Ext*. A greedy Sequential Extraction procedure SE can be formulated as follows:

SE algorithm

Input: set I and data matrices P and Y.

Output: partition $S = \{S_1, S_2, \ldots, S_K\}$ of I in nonintersecting parts (communities) S_k, as well as their characteristics α_k, $k = 1, 2, \ldots, K$, where $K > 0$ is an integer determined as a result of running the algorithm.

Step 1 Set $k = 1$, $D = I$.

Step 2 Apply $(T, alpha) = Ext(D)$ and set $S_k = T$, $\alpha_k = alpha$.

Step 3 Redefine $D = D - S_k$. If $D = \emptyset$ is true, set $K = k$ and stop. Otherwise, define $k = k + 1$ and go to Step 2.

Within this greedy strategy, at its k-th step $(k = 1, 2, \ldots, K)$, we use one more greedy procedure for obtaining a (locally) optimal part $T = S_k$ and its quantitative characteristic α_k. According to this procedure, the set S_k, with its quantitative characteristic c_k, λ_k, at AS, or c_k, λ_{jk} at AN, is found not in one go, but by greedily adding elements of I to S_k one-by-one. The additive structure of criteria (6) and (7) above allows us to express them using contributions to the data scatter, which, to an extent, guides the process, as explained below. Besides its computational simplicity, the sequential extraction approach has some theoretical and practical advantages.

One of the theoretical advantages is a Pythagorean decomposition of the data scatter—this allows scoring the contribution of various elements of found solutions to the data scatter, which is helpful for interpretation (Mirkin 2012). Among practical advantages is the competitiveness of the approach regarding the quality of cluster recovery against other computational procedures (see, for example, experimental results of realizations of the doubly greedy strategy in different situations in Chiang and Mirkin (2010), Mirkin (2012), Nascimento et al. (2015)).

To apply this strategy here, denote the indicator vector of a community T by $t = (t_i)$; its center in the feature space, by

$c = (c_v)$; and the corresponding intensity weights by λ and λ_j depending on the assumption, AS or AN, respectively (the index k is removed because it is not needed here).

Consider three individual items constituting squared error criteria (6) and (7):

(a) The fit between the feature data and the community and its standard point:

$$F_Y(c,t) = \sum_{i,v}(y_{iv} - c_v t_i)^2 \tag{8}$$

(b) The fit between the AS community model and network data:

$$F_{\mathrm{PS}}(\lambda,t) = \sum_{i,j}(p_{ij} - \lambda t_i t_j)^2, \tag{9}$$

(c) The fit between the AN community model and network data:

$$F_{\mathrm{PN}}(\lambda,t) = \sum_{i,j}(p_{ij} - \lambda_j t_i)^2. \tag{10}$$

The total goodness of fit measure is either $f_{\mathrm{AS}} = \rho F_Y + \xi F_{\mathrm{PS}}$ (in criterion (6)) or $f_{\mathrm{AN}} = \rho F_Y + \xi F_{\mathrm{PN}}$ (in criterion (7)). Recall that ρ and ξ are weights to balance two data sources, the features and the links, respectively.

At a specified subset $T \subseteq I$, to minimize criteria (6) and (7) regarding the quantitative characteristics c_v, λ, λ_j, one may separately minimize individual parts (8) over c_v, (9) over λ, and (10) over λ_j because of the additive structure of criteria (6) and (7).

Since each of these three is quadratic regarding the respective numerical characteristic c_v, λ, λ_j, the optimal solutions can be found from the first-order optimality conditions. Let us take the derivatives of F_Y with respect to c_v, F_{PS} with respect to λ, and F_{PN} with respect to λ_j:

$$\frac{\partial F_Y}{\partial c_v} = 2\sum_i (y_{iv} - c_v t_i)(-t_i), \tag{11}$$

$$\frac{\partial F_{\mathrm{PS}}}{\partial \lambda} = 2\sum_{i,j}(p_{ij} - \lambda t_i t_j)(-t_i t_j). \tag{12}$$

$$\frac{\partial F_{\mathrm{PN}}}{\partial \lambda_j} = 2\sum_i (p_{ij} - \lambda_j t_i)(-t_i). \tag{13}$$

Equating each of these to zero would yield, in respect, equations:

$$\sum_i y_{iv} t_i = c_v \sum_i t_i^2, \tag{14}$$

$$\sum_{i,j} p_{ij} t_i t_j = \lambda \sum_i t_i^2 \sum_j t_j^2, \tag{15}$$

and

$$\sum_i p_{ij} t_i = \lambda_j \sum_i t_i^2. \tag{16}$$

Since t_i is 1/0 binary, equality $t_i^2 = t_i$ holds. Thus, $\sum_i t_i^2 = \sum_j t_j^2 = \sum_i t_i = |T|$. Therefore, these equations can be equivalently reformulated as follows:

$$c_v = \frac{\sum_i y_{iv} t_i}{|T|} = \frac{\sum_{i\in T} y_{iv}}{|T|}, \tag{17}$$

$$\lambda = \frac{\sum_{i,j} p_{ij} t_i t_j}{|T|^2} = \frac{\sum_{i,j\in T} p_{ij}}{|T|^2}, \tag{18}$$

and

$$\lambda_j = \frac{\sum_i p_{ij} t_i}{|T|} = \frac{\sum_{i\in T} p_{ij}}{|T|}. \tag{19}$$

In other words, the optimal c_v and λ_j at AN must be central in T: they are within-cluster means of features v and network link columns j. Similarly, at AS, the optimal intensity value λ is equal to the mean within-cluster link value.

Let us now reformulate criteria (8), (9), (10) by opening the parentheses and putting there the found optimal values of c_v, λ, λ_j:

Criterion (8) yields:

$$\begin{aligned} F_Y(c,t) &= \sum_{i,v}(y_{iv} - c_v t_i)^2 = \sum_{i,v}(y_{iv}^2 - 2y_{iv}c_v t_i + c_v^2 t_i) \\ &= \sum_{i,v} y_{iv}^2 - 2\sum_v c_v \sum_i (y_{iv} t_i) + \sum_v c_v^2 |T| \end{aligned}$$

Let us denote the square Y scatter by $Q(Y) = \sum_{i,v} y_{iv}^2$ and take into account that $\sum_i y_{iv} t_i = c_v |T|$ and $\sum_i t_i = |T|$. Then the equation above can be rewritten as

$$F_Y(c,t) = Q(Y) - \sum_v c_v^2 |T| \tag{20}$$

Criterion (9) yields:

$$\begin{aligned} F_{\mathrm{PS}}(\lambda,t) &= \sum_{i,j}(p_{ij} - \lambda t_i t_j)^2 = \sum_{i,j} p_{ij}^2 \\ &\quad - 2\lambda \sum_{i,j} p_{ij} t_i t_j + \lambda^2 \sum_{i,j} t_i t_j. \end{aligned}$$

Let us denote the square P scatter by $Q(P) = \sum_{i,j} p_{ij}^2$ and take into account that $\sum_{i,j} p_{ij} t_i t_j = \lambda \sum_{i,j} t_i t_j$. Then the equation above can be rewritten as

$$F_{\mathrm{PS}}(\lambda, t) = Q(P) - \lambda^2 |T|^2 \tag{21}$$

Similarly, criterion (10) yields:

$$F_{\mathrm{PN}}(\lambda, t) = \sum_{i,j} (p_{ij} - \lambda_j t_i)^2 = \sum_{i,j} p_{ij}^2 - 2 \sum_{i,j} p_{ij} t_i \lambda_j + \sum_j \lambda_j^2 \sum_i t_i.$$

Let us take into account that $\sum_i p_{ij} t_i = \lambda_j \sum_i t_i$. Then the equation above can be rewritten as

$$F_{\mathrm{PN}}(\lambda, t) = Q(P) - \sum_j \lambda_j^2 |T|. \tag{22}$$

Therefore, with the optimal values for c_v, λ, and λ_j determined by T in Eqs. (17), (18), and (19), respectively, criteria (6) and (7) can be equivalently reformulated as

$$f(\lambda, c_v, t_i) = \rho Q(Y) + \xi Q(P) - G \tag{23}$$

where λ is either a scalar or vector, and

$$G(T) = G_s = \rho |T| \sum_v c_v^2 + \xi \lambda \sum_{ij} p_{ij} t_i t_j \tag{24}$$

at the assumption AS, and

$$G(T) = G_n = |T| \left(\rho \sum_v c_v^2 + \xi \sum_j \lambda_j^2 \right) \tag{25}$$

at the assumption AN, where c_v, λ, and λ_j are determined by T according to Eqs. (17), (18), and (19), respectively.

Maximizing criteria $G(T)$ in Eqs. (24) and (25) is equivalent to minimizing the one-cluster least-squares criteria in Eqs. (6) and (7). Therefore, it makes sense to take a look whether $G(T)$ has any meaning of its own.

First of all, we can rewrite Eq. (23) as a Pythagorean decomposition of the combined data scatter $\rho Q(Y) + \xi Q(P)$:

$$\rho Q(Y) + \xi Q(P) = G + f \tag{26}$$

in two parts, the minimized square residuals f and the remaining part G. This decomposition gives meaning to the value of G as the contribution of cluster T to the combined data scatter.

By looking at the formulas for G, we can see that its part related to the feature set, which is the same in both expressions for $G(T)$, (24) and (25), requires maximization of both the cardinality $|T|$ and the squared distance between c and 0, $\sum_v c_v^2$. This means an optimal T should have as many elements as possible and, simultaneously, be as far away from 0 as possible in the feature space. Assuming that the feature data are pre-processed so that the origin is transferred to the center of gravity, the grand mean, the point whose components are the averages of the corresponding features, we may conclude that the cluster T should be both numerous and anomalous. The second item in each of the criteria, G_s (24) and G_n (25), has a similar meaning regarding the network data.

Hence, we refer to our local search algorithm for maximizing (24) or (25) as to the Feature-rich Network Addition Clustering algorithm, FNAC. We use endings, FNACs and FNACn, if necessary, to point out which of criteria (24) and (25), respectively, is maximized. The algorithm finds a cluster T, its center c, and its intensity weight(s) λ (λ_j) by locally maximizing $G(T)$ in the system of neighborhoods defined by the following condition. Given a current T, its neighborhood consists of subsets differing from T by just adding a single entity.

The algorithm starts from a random $i \in I$. This i serves as the seed forming a singleton cluster $T = \{i\}$. This triggers the execution of the base FNAC module. At any current T, this module computes increment $\Delta(j) = G(T + j) - G(T)$ for every element $j \in I - T$ and selects that $j*$ at which $\Delta(j)$ is maximum. If this maximum is positive, then $j*$ is added to T, and the module runs again from thus updated T. If, in contrast, $\Delta(j*) < 0$, the algorithm halts and outputs T, its center c, its link intensity λ (or intensities λ_j), and its contribution to the combined data scatter G. Then the last check is performed: *Seed relevance check* If the seed's removal increases the cluster contribution, this seed is extracted from the cluster.

In its versions FNACs and FNACn, the algorithm FNAC above serves as the core subroutine *Ext* in our community detection algorithm SE above. The algorithm SE involves an internal procedure, $(T, \alpha) = Ext(D)$ where $D \subseteq I$. By using FNAC as the algorithm *Ext* to output the community T along with its parameters c_v and λ / λ_j constituting the α, we obtain a combined algorithm, SEFNAC.

A source code of SEFNACs and SEFNACn and all other supplementary materials, including the real-world data sets, synthetic data generator, etc. are publicly available in https://github.com/Sorooshi/SEFNACs_SEFNACn.

3 Setting of experiments for validation and comparison of the proposed methods

To set a computational experiment, one should specify its constituents:

(1) A set of algorithms under comparison.
(2) A set of data sets at which the algorithms are evaluated and/or compared.
(3) A set of pre-processing methods which are applied to standardize or normalize the data sets.
(4) A set of criteria for assessment of the experimental results.

We describe our settings in separate sections.

3.1 Algorithms under comparison

In addition to our algorithms, SEFNACs and SEFNACn, we take two popular algorithms of the model-based approach, CESNA (Yang et al. 2013) and SIAN (Newman and Clauset 2016), which have been extensively tested in computational experiments. We use the author-made codes of the algorithms which are publicly available. We also tested the algorithm PAICAN from Bojchevski and Günnemann (2018). The results of this algorithm, unfortunately, were always less than satisfactory; therefore, we have excluded the algorithm PAICAN from this paper.

Here are brief descriptions of CESNA and SIAN approaches.

CESNA (Yang et al. 2013) *overview* Given an undirected graph $G(V, E)$ with binary node attribute matrix X, where V is the set of vertices and E is the set of edges, the aim of CESNA is to detect C communities regarding the graph structure and node attributes. The authors define two generative models, one for the graph and the other for attributes, and combine them together. For graph structure they use Eq. (27) to model the probability of an edge between two nodes u and v as follows:

$$\begin{aligned} P_{uv} &= 1 - \exp\left(-\sum_{c=1}^{C} F_{uc}F_{vc}\right) \\ A_{uv} &\sim \text{Bernoulli}(P_{uv}) \end{aligned} \tag{27}$$

where $A \in \{0,1\}^{N\times N}$ denotes the graph adjacency matrix. Unknown function F_{uc} represents the membership of node u to community c, so that the probability is a logistic function of the inner product of F_{uc} and F_{vc} . The presence or absence of an edge uv is governed by a Bernoulli distribution, so that it holds with probability P_{uv} or does not, with probability $1 - P_{uv}$.

Similar model (28) is defined for any binary attribute at nodes:

$$\begin{aligned} Q_{uk} &= \frac{1}{1+\exp(-\sum_c W_{kc}.F_{uc})} \\ X_{uk} &\sim \text{Bernoulli}(Q_{uk}) \end{aligned} \tag{28}$$

here W_{kc} is a real-valued parameter of the logistic model for community c to the *k-th* node attribute.

With the two models above, the problem is to infer values of latent variables F and W by maximizing the likelihood $l(F, W) = \log P(G, X|F, W)$ of the observed data G, X. Here $F = (F_{uc})$ is the node-to-community membership matrix and $W = (W_{kc})$ is the real-valued logistic model parameter for attributes.

Assuming that these two sources of data are conditionally independent, the loglikelihood can be defined as $\log P(G, X|F, W) = L_G + L_X$ where $L_G = \log P(G|F)$ and $L_X = \log P(X|F, W)$. To find F and W maximizing L_G and L_X, which can be computed using Eqs. (27) and (28), the authors adopt projected gradient ascent approach with backtracking line search (Boyd and Vandenberghe 2004).

An author-supplied code for CESNA algorithm can be found at Leskovec and Sosič (2016).

SIAN (Newman and Clauset 2016) *overview* Consider a set of features $\mathbf{x} = \{x_u\}$ at nodes $u = 1, 2, \ldots, n$ and a set of node degrees $\mathbf{d} = \{d_u\}$. Assume, first, that each node u belongs to community s with the probability depending on x_u. and denote all possible combinations of features and communities by $\Gamma = (\gamma_{sx})$. Then the full prior probability of community assignment is $P(\mathbf{s}|\Gamma, \mathbf{x})$. At the next stage, edges between nodes are formed independently at random, with the probability of an edge between nodes u and v being $p_{uv} = d_u d_v \theta_{s_u s_v}$ where θ_{st} is a hyper-parameter.

The task is to fit the model to the observed data by using the maximum likelihood principle. To this end, a binary adjacency matrix $\mathbf{A} = (a_{uv})$, is generated according to the following model:

$$\begin{aligned} P(\mathbf{A}|\Theta, \Gamma, \mathbf{x}) &= \sum_s P(\mathbf{A}|\Theta, \mathbf{s}).P(\mathbf{s}|\Gamma, \mathbf{x}) \\ &= \sum_s \prod_{u<v} p_{uv}^{a_{uv}}(1-p_{uv})^{1-a_{uv}} \prod_u \gamma_{s_u x_u} \end{aligned} \tag{29}$$

Here Θ is a $k \times k$ matrix of elements θ_{st}, and the sum is over all admissible node-to-community assignments. To maximize the function in (29) the authors use the expectation–maximization (EM) algorithm.

An author-supplied code for SIAN algorithm can be found at https://www.nature.com/articles/ncomms11863.

EVA (Citraro and Rossetti 2020) *overview*

Defining a node-attributed graph as $G = (V, E, A)$: where V is the set of nodes, E the set of links, and A is a set of nominal or ordinal attributes such that $A(v)$, for $v \in V$, identifies the set of labels (features) associated with node v. The aim is to discover clusters $C = \{c_1, \ldots, c_n\}$ such that the network links clustering criterion and the feature's homogeneity criterion within each community is maximized.

To this end, authors of EVA (Citraro and Rossetti 2020) model the network links with the popular modularity criterion as follows:

$$Q = \frac{1}{2m}\sum_{v,w}\left[A_{v,w} - \frac{k_v k_w}{2m}\right]\delta(c_v, c_w) \tag{30}$$

where m is the number of links, $A_{v,w}$ is the entry of the adjacency matrix for $v, w \in V$, k_v, k_w are the degrees of node v and node w, respectively. And $\delta(c_v, c_w)$ is an indicator

function taking value 1 when v, w both belong to the same community and 0 otherwise.

The authors model features with a metric called purity. Concretely, for a given community $c \in C$, its purity is the product of the frequencies of the most frequent attribute.

$$P_c = \prod_{a \in A} \frac{\max_{a \in A} \sum_{v \in c} a(v)}{|c|}. \tag{31}$$

where A is the set of features, for $a \in A$ is a feature, $a(v)$ represents an indicator function which is unity iff $a \in A(v)$. Purity ranges in [0,1], and it is maximized when all the nodes within a community share the same attribute. The authors define the purity of a partition as the average of all the community purities:

$$P = \frac{1}{C} \sum_{c \in C} Pc \tag{32}$$

Finally, the authors linearly combine these two criteria as follows:

$$Z = \alpha P + (1 - \alpha)Q \tag{33}$$

where α is a user-defined hyper-parameter to adjust the importance of each source of the two sources of data.

To optimize Eq. (33) two modifications of the Louvain algorithm (Blondel et al. 2008) are adopted.

An author-supplied code for EVA algorithm can be found at https://github.com/GiulioRossetti/EVA.

3.2 Data sets

We use both real-world data sets and synthetic data sets. We describe them in the following subsections.

3.2.1 Real-world data sets

The two out of three algorithms under comparison restrict the features to be categorical, unlike the proposed methods SEFNAC and EVA. Therefore, whenever a data set contains a quantitative feature we convert that feature to a categorical version. A brief overview of the six real-world data sets under consideration can be found in Table 1.

Let us describe them in turn.

Malaria data set

This data set is introduced in Larremore et al. (2013). The nodes are amino acid sequences containing six highly variable regions (HVR) each. The edges are drawn between sequences with similar HVRs 6. In this data set, there are two nominal attributes of nodes:

(1) Cys Labels derived from of a highly variable region HVR6 sequence
(2) Cys-PoLV labels derived from the sequences adjacent to regions HVR 5 and 6

The Cys Labels are considered as the ground truth.

Lawyers data set

The Lawyers data set comes from a network study of corporate law partnership that was carried out in a Northeastern US corporate law firm, referred to as SG & R, 1988–1991, in New England. It is introduced in Lazega (2001) and is available for downloading at https://www.stats.ox.ac.uk/~snijders/siena/Lazega_lawyers_data.htm. There is a friendship network between lawyers in the study. The features in this data set are:

(1) Status (partner, associate),
(2) Gender (man, woman),
(3) Office location (Boston, Hartford, Providence),
(4) Years with the firm,
(5) Age,
(6) Practice (litigation, corporate),

Table 1 Real-world data sets under consideration

Name	Nodes	Edges	Features	Number of communities	Ground truth	References
Malaria HVR6	307	6526	6	2	Cys labels	Larremore et al. (2013)
Lawyers	71	339	18	6	Derived out of office and status features	Lazega (2001), https://www.stats.ox.ac.uk/~snijders/siena/Lazega_lawyers_data.htm
World trade	80	1000	16	5	Structural world system in 1980 features	De Nooy et al. (2004)
Parliament	451	11,646	108	7	Political parties	Bojchevski and Günnemann (2018)
COSN	46	552	16	2	Region	Cross and Parker (2004)
SinaNet	3490	30,282	10	10	Users of same forum	Jia et al. (2017)

Symbols N, E, and F stand for the number of nodes, the number of edges, and the number of node features, respectively

(7) Law school (Harvard or Yale, UCon., Other)

Most features are nominal. Two features, "Years with the firm" and "Age," are quantitative. Authors of the previous studies converted them to the nominal format, which it is accepted in this work as well. The categories of "Years with the firm" are $x <= 10$, $10 < x < 20$, and $x >= 20$; the categories of "Age" are $x <= 40$, $40 < x < 50$, and $x >= 50$.

The combination of Office location and Status is considered as the ground truth (see Table 2).

World-trade data set The World-Trade data set contains data on trade between 80 countries in 1994 (see (De Nooy et al. 2004)). The link weights represent total imports by row countries from column countries, in $ 1000, for the class of commodities designated as "miscellaneous manufactures of metal" to represent high technology products or heavy manufacture. The weights for imports with values less than 1% of the country's total imports are zeroed.

The node attributes are:

(1) Continent (Africa, Asia, Europe, North America, Oceania, South America)
(2) Structural World System Position (Core, Semi-Periphery, Periphery),
(3) Gross Domestic Product per capita in $ (GDP p/c)
(4) Structural World System Position [SWSP] in 1980 according to Smith and White (Core, Semi-Periphery, Periphery, N.A.I) N.A.I: stands for not available information for countries in which their ecumenical information was not available due to various reasons such as war or dictatorship.

The Structural World System Position in 1980 according to De Nooy et al. (2004) is considered as the ground truth.

The GDP p/c feature is converted into a three-category nominal feature manually, according to the minima of its histogram. The categories are defined as follows: "Poor" category is for the GDP p/c less than $4406.9; "Mid-Range" category is for the GDP p/c greater than $4406.9 but not greater than $21574.5; and "Wealthy" category corresponds to the GDP p/c greater than $21574.5.

These features are reviewed in Table 3. Before applying SEFNAC, all attribute categories are converted into 0/1 dummy variables which are considered quantitative.

Parliament data set

In the Parliament data set, introduced in Bojchevski and Günnemann (2018), nodes correspond to members of the French Parliament. An edge is drawn if the corresponding MPs sign a bill together. The features are the constituency of MPs and their political party, as it is described by the authors. The latter is considered the ground truth (see Table 4).

Consulting Organisational Social Network (COSN) data set

The Consulting Organisational Social Network (COSN) data set is introduced in Cross and Parker (2004). Nodes in

Table 2 Features in Lawyers data set

No.	Feature	Type	Categories	*N*	*E*	*F*
1	Status	Nominal	Partner, associate			
2	Gender	Nominal	Male, female			
3	Office	Nominal	Boston, Hartford, Providence			
4	Years with firm	Categorized	$x <= 10, 10 < x < 20, x >= 20$	71	399	18
5	Age	Categorized	$x <= 40, 40 < x < 50, x >= 50$			
6	Practice	Nominal	Litigation, Corporate			
7	Law school	Nominal	Harvard, Yale, UCon.			

Symbols *N*, *E*, and *F* denote the number of nodes, the number of edges, and the number of node features, respectively

Table 3 Features in World Trade data set

No.	Feature	Type	Categories	*N*	*E*	*F*
1	Continent	Nominal	Africa, Asia, Europe, North America, Oceania, South America			
2	SWSP in 1994	Nominal	Core, Semi-periphery, Periphery	80	1000	16
3	GDP categories	Nominal	Poor, mid-range, wealthy			
4	SWSP in 1980 according to Smith and White	Nominal	Core, semi-periphery, periphery, N.I.A			

SWSP stands for Structural World System Position; GDP, for Gross Domestic Product per capita. Symbols *N*, *E*, and *F* show the number of nodes, the number of edges, and the number of node features, respectively

this network correspond to employees in a consulting company. The (asymmetric) edges are formed in accordance with their replies to this question: "Please indicate how often you have turned to this person for information or advice on work-related topics in the past three months." The answers are coded by 0 (I Do Not Know This Person), 1 (Never), 2 (Seldom), 3 (Sometimes), 4 (Often), and 5 (Very Often). Either of these 6 numerals is the weight of all the corresponding edges.

Nodes in this network have the following attributes:

(1) Organizational level (Research Assistant, Junior Consultant, Senior Consultant, Managing Consultant, Partner),
(2) Gender (Male, Female),
(3) Region (Europe, USA),
(4) Location (Boston, London, Paris, Rome, Madrid, Oslo, Copenhagen).

The Region feature is considered as the ground truth. A description of the data is shown in Table 5.

SinaNet data set (Jia et al. 2017) This data set is a microblog user relationship network extracted from the Sina-microblog website, http://www.weibo.com. The authors at first selected 100 VIP Sina-microblog users distributed in 10 significant forums including finance and economics, literature and arts, Etc. Starting from 100 VIP Sina-microblog users, they extract followers/followings of these users and their published microblogs. Using the depth-first search strategy, they extract three layers of user relationships and obtained 8452 users, 147,653 user relationships, and 5.5 million microblogs in total. They merged all microblogs that a user published to characterize the user's interests. After removing silent users that published less than 5000 words, we left 3490 users and 30282 relationships. By using words' frequency of the merged blogs of a user to describe the user's interest, the feature space's dimension would be too high to be processed. Therefore, they use users' topic distribution in the ten forums obtained by the LDA topic model, which describes users' interests. Thus, besides the follower/following relationships between pairs of users, we have ten-dimensional numerical attributes to describe each user's interests (See Table 6).

3.2.2 Generating synthetic data sets

In this section, we describe how we generate synthetic feature-rich data sets with an innate cluster structure by separately generating:

- Network;
- Categorical features;
- Quantitative features.

Each of these is put in a separate subsection.

Table 4 The Parliament data set

No.	Feature	Type	Categories	*N*	*E*	*F*
1	Constituency	Nominal	MPs constituency	451	11,464	108

Symbols *N*, *E*, and *F* show the number of nodes, the number of edges, and the number of node features, respectively

Table 5 The Consulting Organisational Social Network (COSN) data set

No.	Feature	Type	Categories	*N*	*E*	*F*
1	Organizational level	Nominal	Assistant, junior, consultant, senior consultant, managing consultant, partner			
2	Gender	Nominal	Male, female	46	552	16
3	Region	Nominal	Europe, USA,			
4	Location	Nominal	Boston, London, Paris, Rome, Madrid, Oslo, Copenhagen			

N, *E*, and *F* stand for the number of nodes, the number of edges, and the number of node features, respectively

Table 6 The SinaNet data set

No.	Feature	Type	Categories	*N*	*E*	*F*
1	LDA topic model	Nominal	Dictionary	3490	30,282	10

Symbols *N*, *E*, and *F* show the number of nodes, the number of edges, and the number of node features, respectively

Generating networks

First, the number of nodes, N, and the number of communities K are specified. Then cardinalities of communities are defined randomly and uniformly, up to a constraint that no community has less than a pre-specified number of nodes (in our experiments, this is set to 30), so that probabilistic approaches are applicable, and the total number of nodes in all the communities is equal to N. Using the random option for cardinalities of the communities to be generated is motivated by these reasons:

- Real-world cluster structures in general do not show any specific regularity in cluster structures: some tend to see one or two big clusters among a multitude of very small ones; some tend to claim a natural power law for cluster cardinalities; still, some prefer relatively balanced cluster sizes, especially in human-made systems.
- In our experiences, cardinalities of hidden clusters play no role in the quality of cluster recovery by conventional clustering algorithms.
- This also reduces to zero the number of parameters to control.

We consider two settings for N: (a) $N = 200$, for a small size network, and (b) $N = 1000$, for a medium-size network. We postpone analysis of larger networks for another paper.

Given the community sizes, we populate them with nodes, that are specified just by indices in their natural order. Say, if there are two clusters with 60 and 40 elements, respectively, to be defined, then indices from 1 through 60 go into cluster 1, and those numbered from 61 to 100 go into cluster 2. Then we specify two probability values, p and q.

For every pair of nodes from the same community, an edge is drawn between them with the probability p, independently of other edges. Similarly, for every pair of nodes from different communities, an edge is drawn between them with the probability q, independently of other edges.

Figure 2 illustrates similarity matrices for generated networks at $p = 0.7, 0.9$ and $q = 0.4, 0.6$. The upper pane in the figure visualizes a network with 200 nodes and five communities, whereas the lower pane presents 15 communities at 1000 nodes.

Generating quantitative features

To model quantitative features, we use conventional Gaussian distributions as within-cluster density functions. We apply design proposed in Kovaleva and Mirkin (2015). Each cluster is generated from a Gaussian distribution whose covariance matrix is diagonal with diagonal values that are random and uniform in the range [0.05, 0.1]—they specify the cluster's spread. Each component of the cluster center is generated randomly and uniformly from the range $\alpha[-1, +1]$. Here $\alpha > 0$ controls the cluster intermix: the smaller the α, the greater the chance that points from a cluster fall within the spreads of other clusters. Figure 3 illustrates examples of the generated data sets for $\alpha = 0.7$ and $\alpha = 0.9$. The upper pane in the figure visualizes a feature-rich network with 200 nodes and five communities, whereas the lower pane presents 15 communities at 1000 nodes.

In addition to cluster intermix, the possibility of presence of noise in data is also taken into account. A uniformly random noise feature from an interval defined by the maximum and minimum values is generated and added to the feature data. In this way, 50% of the number of original features are inserted as noise features.

Generating categorical features

To model categorical features, the number of subcategories for each category is randomly chosen from the set $\{2, 3, \ldots, L\}$ where $L = 10$ for small-size networks and $L = 15$ for the medium-size networks. Then, given the number of communities, K, and the numbers of entities, N_k for $(k = 1, \ldots, K)$; the cluster centers are generated randomly so that no two centers may coincide at more than 50% of features.

Once a center of k-th cluster, $c_k = (c_{kv})$, is specified, N_k entities of this cluster are generated as follows. Given a pre-specified threshold of intermix, ϵ between 0 and 1, for every pair (i, v), $i = 1 : N_k$; $v = 1 : V$, a uniformly random real number r between 0 and 1 is generated. If $r > \epsilon$, the entry x_{iv} is set to be equal to c_{kv}; otherwise, x_{iv} is taken randomly from the set of subcategories specified for feature v.

Consequently, all entities in cluster k-th coincide with its center, up to rare errors if ϵ is large enough. The smaller the epsilon, the more diverse, and thus intermixed, would be the generated entities.

To generate a feature-rich network combining categorical and quantitative features, we divide the number of features in two approximately equal parts, one to consist of quantitative features, the other, of categorical features. Each part is filled in independently, according to the schemes described above.

3.3 Data pre-processing

Results of our SEFNAC method depend on the way the data are standardized. Unfortunately, no theoretical foundations have been developed so far for the issues of data standardization. We describe here two popular methods of standardization for feature data and two standardization methods for network data.

For features, we consider the following standardization methods:

(1) Z-scoring: each of the features is centered by subtraction of its mean from its values and then normalized by dividing over its standard deviation;
(2) Range standardization: each of the features is centered by subtraction of its mean from all its values and then

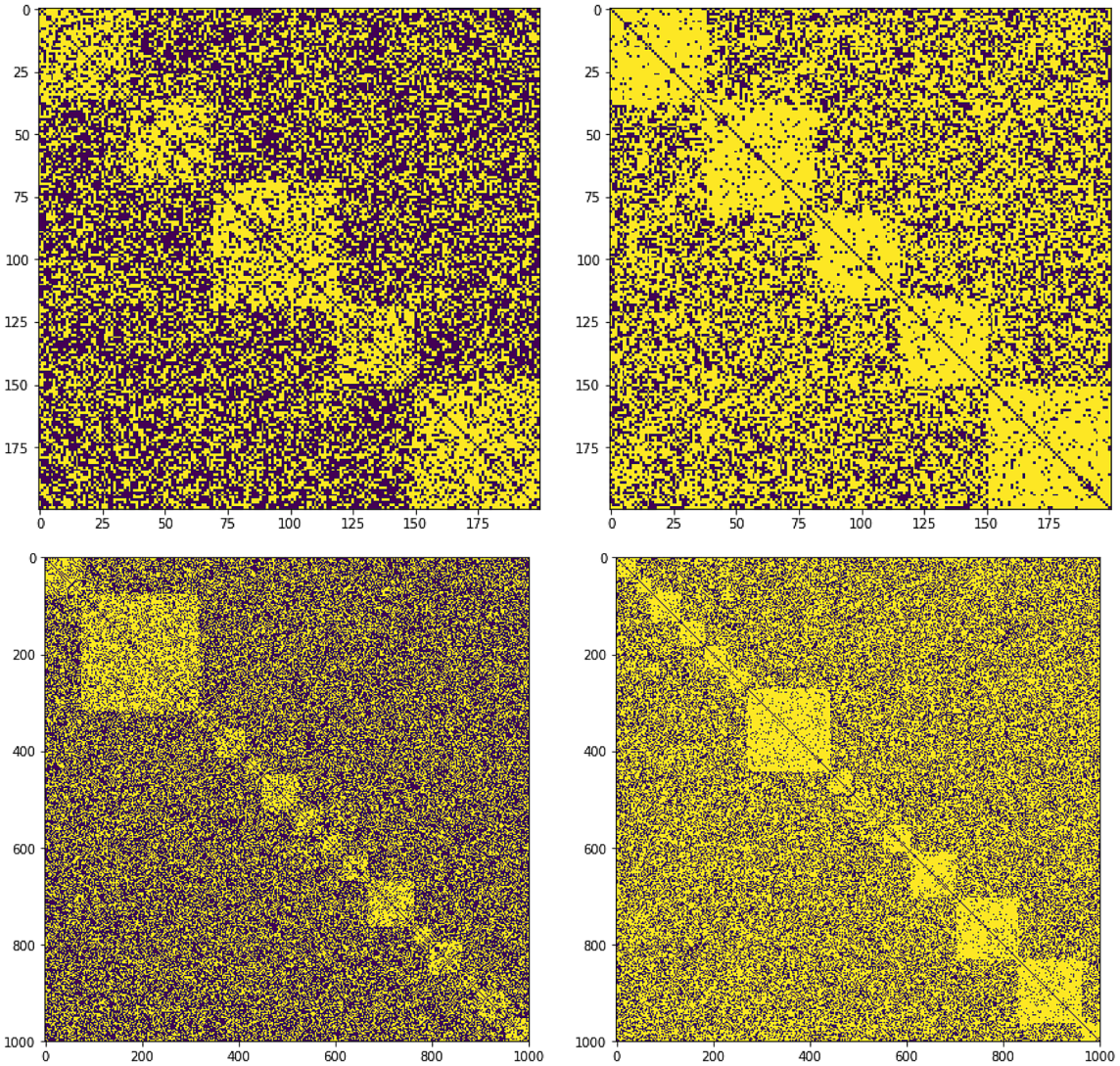


Fig. 2 Samples of synthetically generated network matrices (white pixels represent unities, and dark ones, zeros. The number of nodes is N and the number of communities is K. Values p, q are the probabilities of drawing edges within-community and between communities, respectively. Specifically, $p = 0.7, q = 0.4, N = 200, K = 5$ at **a** $p = 0.9, q = 0.6, N = 200, K = 5$ at **b** $p = 0.7, q = 0.4, N = 1000, K = 15$ at **c** and $p = 0.9, q = 0.6, N = 1000, K = 15$ at **d**

normalized by dividing over its range, that is the difference between its maximum and minimum.

For network data, the two following link normalization options are considered:

(1) Modularity (Newman 2006): Given an $N \times N$ similarity matrix $P = (p_{ij})$, compute summary values $p_{i+} = \sum_{j=1}^{N} p_{ij}$, $p_{+j} = \sum_{i=1}^{N} p_{ij}$, $p_{++} = \sum_{i,j=1}^{N} p_{ij}$ and define random interaction values $r_{ij} = p_{i+}p_{+j}/p_{++}$. Change all p_{ij} for $p_{ij} - r_{ij}$ by removing the random interactions.
(2) Uniform shift (Mirkin 2012): Compute the mean $\pi = \sum_{i,j=1}^{N} p_{ij}/N^2$; change all p_{ij} for $p_{ij} - \pi$.

Each of the two normalizations leads to a similarity matrix in which the mean is zero.

3.4 Evaluation criteria

To compare results found by clustering algorithms, we use a popular metric of similarity between partitions: the Adjusted Rand Index (ARI) (Hubert and Arabie 1985). Also, we used the Normalized Mutual Information (NMI) index (Cover and Thomas 2012; Strehl and Ghosh 2002). The latter has shown results that are very similar to those observed with the ARI; therefore, we decided to omit NMI from the resulting tables.

To define ARI, the so-called contingency table is used.

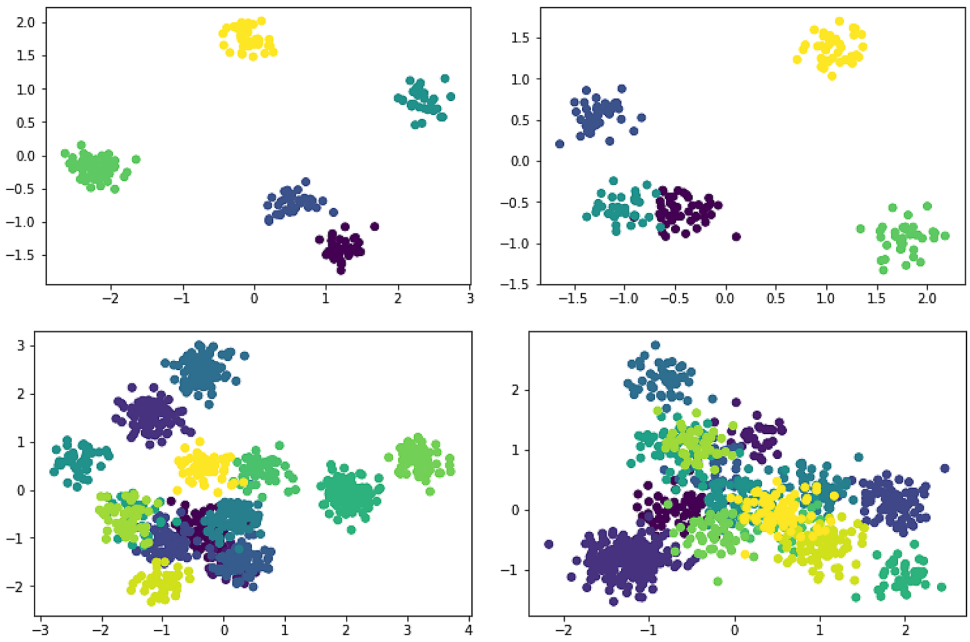


Fig. 3 Samples of synthetically generated clusters at quantitative features; N is the number of nodes, K is the number of communities, and α is the parameter of cluster intermix. The parameter values are: $\alpha = 0.9, N = 200, K = 5$ at (**a**); $\alpha = 0.7, N = 200, K = 5$ at (**b**); $\alpha = 0.9, N = 1000, K = 15$ at (**c**); and $\alpha = 0.7, N = 1000, K = 15$ at (**d**)

Given two partitions of the entity set, $S = \{S_1, S_2, \ldots, S_K\}$ and $T = \{T1, T_2, \ldots, T_L\}$, the contingency table is a two-way table, whose rows correspond to parts S_k $(k = 1, 2, \ldots, K)$ of S, and columns, to parts $l = 1, 2, \ldots, L$ of T, so that its $((k, l))$-th entry is $n_{kl} = |S_k \cap T_l|$, with the so-called marginal row and marginal column defined as $a_k = \sum_{l=1}^{L} n_{kl} = |S_k|$ and $b_l = \sum_{k=1}^{K} n_{kl} = |T_l|$.

The Adjusted Rand Index is defined as:

$$\mathrm{ARI}(S,T) = \frac{\sum_{k,l}\binom{n_{kl}}{2} - \left[\sum_k \binom{a_k}{2}\sum_l \binom{b_l}{2}\right]/\binom{N}{2}}{\frac{1}{2}\left[\sum_k \binom{a_k}{2} + \sum_l \binom{b_l}{2}\right] - \left[\sum_k \binom{a_k}{2}\sum_l \binom{b_l}{2}\right]/\binom{N}{2}} \tag{34}$$

The closer the value of ARI to unity, the better is the match between the two partitions; ARI=1.0 shows that $S = T$. If one of the partitions consists of just one part, the set I itself, then ARI=0. Cases at which ARI is negative may occur too, but these authors have observed them only at specially defined, "dual," pairs of partitions (see in Kovaleva and Mirkin 2015).

4 Experimental comparison of the methods under consideration

4.1 Comparison of the methods over real-world data sets

In this section we compare the performance of our proposed SEFNAC methods with state-of-the-art algorithms, EVA, SIAN, and CESNA, at the six real-world data sets described above in subsection (3.2). All the algorithms are run starting from random configurations ten times at each of the data sets; we report the average ARI values and corresponding standard deviations.

Results of both SEFNACs and SEFNACn depend on data standardization. We chose those pre-processing methods that lead, on average, to the larger ARI values. Table 7 explains the applied pre-processing methods.

The comparison of all the algorithms under consideration over Real-World data sets is recorded in Table 8. Unlike synthetic data sets, SIAN shows better performance on

Table 7 Standardization options chosen for SEFNAC methods on the real-world data sets

Data set	SEFNACs		SEFNACn	
	Y	P	Y	P
Malaria HVR6	Z-scoring	Uniform	Z-scoring	Uniform
Lawyers	Range std.	Uniform	Z-scoring	Uniform
World trade	Range std.	Range std.	Range std.	Range std.
Parliament	Z-scoring	Modularity	Range std.	Uniform
COSN	Z-scoring	No Pre-Process.	Z-scoring	Uniform
SinaNet	Z-scoring	Modularity	Range std.	Uniform

Table 8 Comparison of CESNA, SIAN, DMoN, SEFNACs, SEFNACn on real-world data sets; average values and standard deviations of ARI are presented over 10 random initializations

Data set	CESNA	SIAN	EVA	SEFNACs	SEFNACn
HRV6	0.20 (0.00)	0.39 (0.29)	0.036 (0.004)	**0.49 (0.11)**	0.42 (0.04)
Lawyers	0.28 (0.00)	0.59 (0.04)	0.159 (0.028)	**0.60 (0.09)**	0.59 (0.09)
World trade	0.13 (0.00)	0.10 (0.01))	− 0.003 (0.000)	0.29 (0.10)	**0.41 (0.10)**
Parliament	0.25 (0.00)	**0.79 (0.12)**	0.005 (0.001)	0.28 (0.01)	0.47 (0.09)
COSN	0.44 (0.00)	**0.75 (0.00)**	0.004 (0.000)	0.72 (0.02)	0.54 (0.04)
SinaNet	0.09 (0.00)	0.17 (0.02)	0.001 (0.002)	0.21 (0.03)	**0.25 (0.02)**

The best results are highlighted in bold-face, and second ones are under-lined

real-world data sets. It is the winner for the Parliament and the COSN data sets. The SEFNACs wins the competition on HVR and Lawyers data sets. Finally, SEFNACn wins the competition on World-trade and SinaNet data sets.

4.2 Comparison of the methods over synthetic data sets with categorical features

The comparison of the algorithms over the small-size network with categorical features is reported in Table 9. One can clearly see that the upper six rows' results drastically differ from those at the two last rows. These latter rows correspond to most difficult combination of probabilities, $p = 0.7$ and $q = 0.6$. Indeed, in this case, the probability of between-community edges, $q = 0.6$, is almost as high as the probability of within-community edges, $p = 0.7$.

EVA performs indeed poorly. This poor performance could have two reasons as follows. First, the Modularity criterion, which is adopted to model the networks, usually assigns more weights to less connected nodes. Recall that our generated network data are indeed dense, and nodes have significant between community links. Thus Modularity loses its efficiency. The second reason is related to the purity of features. To be more precise, the authors assume that features of a community's nodes are identical, and, in our opinion, this is a very optimistic assumption, which is not the case in our generated data set because of the parameter ϵ we used to control the homogeneity of features.

Unfortunately, SIAN also performs poorly, especially in the two most challenging cases, in which ARI is almost zero. Such a performance might happen because of the networks' sparsity assumption in Newman and Clauset (2016). Another reason for this poor performance might be the convergence, i.e., stocking, in a local optimum.

The results by CESNA and SEFNACs are relatively similar in the upper six cases. In the two most complex cases, the results by CESNA dramatically fall, whereas those by SEFNACs remain relatively high, being the best out of the five algorithms under comparison.

It ought to mention that, on average, SEFNACn's performance is also acceptable. Furthermore, recalling its faster execution time than SEFNACs, we can consider it a preferable algorithm for larger data sets.

The results in Table 10 show that the performance of SEFNACs significantly improves when the size of networks increases. CESNA also works well. It obtains decent results in 5 out of 8 settings. However, in most cases, SEFNACs achieve high ARI values. Noteworthy to mention that the SEFNAC methods automatically determine the number of clusters. SEFNACn also achieves good results on almost all the settings except for the last one.

The poor performance of SIAN on the medium-size networks could be explained by either (a) the convergence issues or (b) issues at computing the prior probabilities,

Table 9 Comparison of CESNA, SIAN, SEFNACs, and SEFNACn over small-size synthetic data sets with categorical attributes

p, q, α	CESNA	SIAN	EVA	SEFNACs	SEFNACn
0.9, 0.3, 0.9	**1.00 (0.00)**	0.554 (0.285)	0.185 (0.046)	0.994 (0.008)	**1.000 (0.000)**
0.9, 0.3, 0.7	0.948 (0.105)	0.479 (0.289)	0.211 (0.053)	0.974 (0.024)	**1.000 (0.000)**
0.9, 0.6, 0.9	0.934 (0.075)	0.320 (0.255)	0.287 (0.060)	0.965 (0.013)	**0.972 (0.053)**
0.9, 0.6, 0.7	0.902 (0.063)	0.110 (0.138)	0.179 (0.050)	0.750 (0.117)	**0.906 (0.048)**
0.7, 0.3, 0.9	0.965 (0.078)	0.553 (0.157)	0.126 (0.039)	0.975 (0.018)	**0.992 (0.007)**
0.7, 0.3, 0.7	0.890 (0.138)	0.508 (0.211)	0.126 (0.025)	0.870 (0.067)	**0.981 (0.020)**
0.7, 0.6, 0.9	0.506 (0.101)	0.047 (0.087)	0.019 (0.005)	**0.896 (0.067)**	0.821 (0.037)
0.7, 0.6, 0.7	0.202 (0.081)	0.030 (0.040)	0.010 (0.008)	**0.605 (0.091)**	0.223 (0.073)

The best results are highlighted in bold-face. The average ARI index and its standard deviation values over 10 different data sets are presented

Table 10 Comparison of CESNA, SIAN, SEFNACs, and SEFNACn over medium-size synthetic data sets with categorical attributes

p, q, α	CESNA	SIAN	EVA	SEFNACs	SEFNACn
0.9, 0.3, 0.9	0.894 (0.053)	0.000 (0.000)	0.126 (0.034)	**1.000 (0.000)**	**1.000 (0.000)**
0.9, 0.3, 0.7	0.849 (0.076)	0.000 (0.000)	0.077 (0.023)	**0.996 (0.005)**	0.697 (0.345)
0.9, 0.6, 0.9	0.632 (0.058)	0.000 (0.000)	0.174 (0.037)	**0.998 (0.002)**	0.941 (0.163)
0.9, 0.6, 0.7	0.474 (0.089)	0.000 (0.000)	0.103 (0.030)	**0.959 (0.032)**	0.736 (0.210)
0.7, 0.3, 0.9	0.764 (0.068)	0.026 (0.077)	0.081 (0.038)	**1.000 (0.001)**	0.919 (0.229)
0.7, 0.3, 0.7	0.715 (0.128)	0.000 (0.000)	0.064 (0.015)	**0.993 (0.002)**	0.850 (0.204)
0.7, 0.6, 0.9	0.060 (0.024)	0.000 (0.000)	0.003 (0.002)	**0.998 (0.001)**	0.879 (0.085)
0.7, 0.6, 0.7	0.016 (0.008)	0.000 (0.000)	0.001 (0.001)	**0.909 (0.035)**	0.329 (0.104)

The best results are highlighted in bold-face. The average ARI index and its standard deviation values over 10 different data sets are presented

which might be inconsistent with the assumption of the sparsity of the networks in Newman and Clauset (2016).

Similar to the small-size networks, we observe abysmal performance from EVA. As mentioned earlier, this could be either due to the dense generated networks or the nonhomogeneous features, which contradict the authors' assumption of the network links and node features, respectively.

5 Experimental validation of SEFNAC methods

5.1 Choosing the data standardization options

In Sect. (3.3), we considered two options for feature data standardization: Z-scoring, further denoted as Z-score, and Range standardization, further denoted as Range. Similarly, we considered two options for network data standardization: Modularity transformation, referred to as Modular and Uniform shift, further referred to as Uniform. This section aims to assign SEFNACs and SEFNACn a unique set of data standardization options for all the experiments with synthetic data sets.

To choose the most suitable combinations, we applied all possible combinations of the pre-processing techniques on the synthetic data sets with (a) only quantitative features, (b) only categorical features, (c) mixed-scaled features. However, for the sake of brevity, we only provided results of the latter case.

Table 11 presents cluster-recovery results for SEFNACs at synthetic small-size networks with the combination of quantitative and categorical features under different data standardization techniques. Similarly, Table 12 gives a similar account for SEFNACn.

Table 11 shows that the Z-score-Uniform standardization gives an edge over the other standardizations for SEFNACs.

Regarding Table 12, one should conclude that the pair Z-score-Uniform standardization on average yields superior results for the SEFNACn, too. This is especially clear at the worst combination of parameters, $(p, q, \alpha/\epsilon) = (0.7, 0.6, 0.7)$. Comparing the two tables, one may reasonably conclude that SEFNACs work better than SEFNACn in this setting.

Summarizing all the computation on the early mentioned synthetic data sets (only quantitative feature scales, only categorical feature scales, combined feature scales),

Table 11 The performance of SEFNACs on small-size networks with combined quantitative and categorical features at the nodes for different data standardization options

$p, q, \alpha \vert \epsilon$	Z-score-modular		Z-score-uniform		Range-modular		Range-uniform	
Setting p, q, α	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	0.972 (0.065)	4.900 (0.300)	**0.996 (0.011)**	**5.000 (0.000)**	0.158 (0.150)	2.300 (0.900)	0.233 (0.218)	2.800 (1.077)
0.9, 0.3, 0.7	**0.982 (0.029)**	**5.000 (0.000)**	**0.982 (0.025)**	**5.000 (0.000)**	0.023 (0.038)	1.700 (1.004)	0.135 (0.252)	2.300 (1.486)
0.9, 0.6, 0.9	**0.917 (0.091)**	**4.600 (0.489)**	0.914 (0.093)	4.600 (0.489)	0.105 (0.097)	3.100 (1.220)	0.093 (0.109)	2.500 (1.360)
0.9, 0.6, 0.7	0.825 (0.119)	4.400 (0.489)	**0.865 (0.139)**	**4.800 (0.600)**	0.030 (0.044)	3.000 (1.788)	0.009 (0.024)	3.200 (1.989)
0.7, 0.3, 0.9	0.991 (0.020)	5.000 (0.000)	**0.993 (0.016)**	**5.000 (0.000)**	0.225 (0.228)	3.100 (0.943)	0.268 (0.293)	2.700 (1.486)
0.7, 0.3, 0.7	0.919 (0.128)	4.900 (0.300)	**0.943 (0.105)**	**4.900 (0.300)**	0.171 (0.259)	2.000 (1.000)	0.171 (0.299)	2.400 (1.356)
0.7, 0.6, 0.9	**0.736 (0.197)**	**3.800 (0.871)**	**0.736 (0.197)**	**3.800 (0.871)**	0.213 (0.155)	3.200 (0.871)	0.137 (0.145)	4.000 (1.549)
0.7, 0.6, 0.7	0.607 (0.137)	3.900 (0.700)	**0.665 (0.143)**	**4.300 (1.100)**	0.028 (0.030)	2.300 (1.100)	0.011 (0.015)	2.900 (1.757)

The best results are highlighted in bold-face. The average ARI index and its standard deviation values are presented over 10 different data sets

we can conclude that: as a rule, we combine Z-scoring with the Uniform transformation. There are only two exceptions from this rule: (a) at quantitative features only, we combine Z-scoring and Modularity standardization for SEFNACs, (b) at categorical features only, we combine Range standardization with the Uniform transformation for SEFNACn.

5.2 Experimental results for SEFNAC algorithms at various feature scales

5.2.1 SEFNAC at synthetic networks with categorical features at the nodes

Following Tables 13 and 14 present results of SEFNACs

Table 12 The performance of SEFNACn on small-size networks with combined quantitative and categorical features at the nodes for different data standardization options

$p, q, \alpha \vert \epsilon$	Z-score-modular		Z-score-uniform		Range-modular		Range-uniform	
Setting p, q, α	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	**1.000 (0.000)**	**5.000 (0.000)**	**1.000 (0.000)**	**5.000 (0.000)**	0.940 (0.178)	4.700 (0.900)	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.3, 0.7	**1.000 (0.000)**	**5.000 (0.000)**	**1.000 (0.000)**	**5.000 (0.000)**	**1.000 (0.000)**	**5.000 (0.000)**	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.6, 0.9	0.862 (0.107)	4.800 (0.872)	0.841 (0.084)	4.800 (0.748)	0.788 (0.155)	5.500 (0.500)	**0.894 (0.060)**	**5.800 (0.600)**
0.9, 0.6, 0.7	0.836 (0.086)	5.200 (0.980)	**0.839 (0.115)**	**5.300 (1.187)**	0.667 (0.174)	6.400 (0.800)	0.798 (0.105)	6.100 (0.943)
0.7, 0.3, 0.9	0.973 (0.042)	4.900 (0.300)	0.961 (0.064)	4.800 (0.400)	0.877 (0.147)	4.700 (0.458)	**0.983 (0.009)**	**5.000 (0.000)**
0.7, 0.3, 0.7	0.929 (0.079)	4.800 (0.600)	0.952 (0.061)	4.800 (0.400)	0.912 (0.124)	5.100 (0.831)	**0.965 (0.011)**	**5.400 (0.490)**
0.7, 0.6, 0.9	0.707 (0.109)	5.500 (0.671)	**0.745 (0.111)**	**6.000 (1.000)**	0.014 (0.017)	9.500 (1.025)	0.017 (0.011)	11.400 (1.200)
0.7, 0.6, 0.7	0.548 (0.074)	6.600 (1.281)	**0.587 (0.115)**	**7.300 (1.616)**	0.010 (0.017)	9.800 (0.980)	0.031 (0.018)	11.900 (0.539)

The best results are highlighted in bold-face. The average ARI index and its standard deviation values are presented over 10 different data sets

Table 13 The performance of SEFNACs and SEFNACn on small-size networks with categorical features at the nodes

p, q, ϵ	SEFNACs		SEFNACn	
Setting p, q, ϵ	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	0.996 (0.011)	5.000 (0.000)	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.3, 0.7	0.974 (0.038)	5.000 (0.000)	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.6, 0.9	0.963 (0.019)	5.200 (0.400)	**0.972 (0.053)**	**5.000 (0.447)**
0.9, 0.6, 0.7	0.739 (0.113)	7.800 (0.600)	**0.906 (0.048)**	**6.400 (0.663)**
0.7, 0.3, 0.9	0.978 (0.015)	5.200 (0.400)	**0.992 (0.007)**	**5.000 (0.000)**
0.7, 0.3, 0.7	0.865 (0.070)	6.500 (1.025)	**0.981 (0.020)**	**5.000 (0.000)**
0.7, 0.6, 0.9	**0.898 (0.069)**	**6.500 (0.500)**	0.821 (0.037)	8.200 (1.249)
0.7, 0.6, 0.7	**0.595 (0.104)**	**8.000 (1.095)**	0.223 (0.073)	10.800 (1.166)

The average ARI index and its standard deviation values are presented over 10 different data sets

Table 14 The performance of the SEFNACs and SEFNACn on medium-size networks with categorical features at the nodes

p, q, ϵ	SEFNACs		SEFNACn	
Setting p, q, ϵ	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	**1.000 (0.000)**	**15.000 (0.000)**	**1.000 (0.000)**	**15.000 (0.000)**
0.9, 0.3, 0.7	**0.996 (0.005)**	**15.000 (0.000)**	0.697 (0.345)	10.800 (4.707)
0.9, 0.6, 0.9	**0.998 (0.002)**	**15.000 (0.000)**	0.941 (0.163)	14.500 (1.500)
0.9, 0.6, 0.7	**0.959 (0.032)**	**16.500 (1.025)**	0.736 (0.210)	13.500 (1.962)
0.7, 0.3, 0.9	**1.000 (0.001)**	**15.000 (0.000)**	0.919 (0.229)	13.700 (3.257)
0.7, 0.3, 0.7	**0.993 (0.002)**	**15.700 (0.458)**	0.850 (0.204)	12.800 (2.482)
0.7, 0.6, 0.9	**0.998 (0.001)**	**15.100 (0.300)**	0.879 (0.085)	13.900 (1.814)
0.7, 0.6, 0.7	**0.909 (0.035)**	**20.300 (1.269)**	0.329 (0.104)	15.500 (1.118)

The best results are highlighted in bold-face. The average ARI index and its standard deviation values are presented over 10 different data sets

and SEFNACn methods at small-size and medium-size synthetic networks, respectively, with categorical features only. The chosen data standardization options are described at the end of the previous section.

One can see that SEFNACs overwhelmingly wins at the medium-sized feature-rich networks showing outstanding results even in the worst scenario of parameter setting $(p, q, \epsilon) = (0.7, 0.6, 0.7)$. The level of reproducing the numbers of communities by this algorithm looks impressive indeed according to Table 14. On the contrary, SEFNACn leads to better results at the small-sized data at almost all parameter settings except for the last two.

5.2.2 SEFNAC at synthetic networks with quantitative features at the nodes

Following Tables 15 and 16 present results of SEFNACs and SEFNACn methods at small-size and medium-size synthetic networks, respectively, with quantitative features only. The chosen data standardization options are described at the end of Sect. (5.1).

As one can see in Table 15, the SEFNACs is the winner. However, the obtained results of the SEFNACn are good too.

Considering the results presented in Table 16, SEFNACs is the sole winner.

Let us check if the performances of the algorithms change when noise features are inserted. The results are presented in Tables 17 and 18.

Similarly, the SEFNACs are the winner, although the SEFNACn usually shows good results too. The setting with noise features gives one more effect. We can see that changing q to 0.6 at $p = 0.9$ reduces the clusters' reproducibility greater than the decrease of p to 0.7 at a smaller q-rate of 0.3.

5.2.3 SEFNAC at synthetic data sets combining quantitative and categorical features

Following Tables 19 and 20 present results of SEFNACs and SEFNACn methods at small-size and medium-size synthetic networks, respectively, with combined quantitative and categorical features. The chosen data standardization options are described at the end of Sect. (5.1).

Table 15 The performance of the SEFNACs and SEFNACn on small-size networks with quantitative features at the nodes

p, q, α	SEFNACs		SEFNACn	
Setting p, q, α	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	**1.000 (0.000)**	**5.000 (0.000)**	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.3, 0.7	0.999 (0.004)	5.000 (0.000)	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.6, 0.9	0.935 (0.109)	4.900 (0.539)	**0.963 (0.051)**	**4.900 (0.300)**
0.9, 0.6, 0.7	**0.989 (0.018)**	**5.000 (0.000)**	0.938 (0.070)	4.900 (0.539)
0.7, 0.3, 0.9	**0.990 (0.025)**	**5.000 (0.000)**	0.971 (0.052)	4.800 (0.400)
0.7, 0.3, 0.7	**0.997 (0.010)**	**5.000 (0.000)**	0.968 (0.057)	4.800 (0.400)
0.7, 0.6, 0.9	**0.787 (0.161)**	**5.100 (1.758)**	0.783 (0.095)	5.400 (2.154)
0.7, 0.6, 0.7	0.705 (0.182)	5.100 (1.300)	**0.713 (0.113)**	**6.200 (1.249)**

The best results are highlighted in bold-face. The average ARI index and its standard deviation values are presented over 10 different data sets

Table 16 The performance of SEFNACs and SEFNACn on medium-size networks with quantitative features at the nodes

p, q, α	SEFNACs		SEFNACn	
Setting p, q, α	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	**1.000 (0.000)**	**15.000 (0.000)**	0.979 (0.032)	14.300 (1.005)
0.9, 0.3, 0.7	**1.000 (0.000)**	**15.00 (0.000)**	0.873 (0.200)	12.200 (3.789)
0.9, 0.6, 0.9	**0.860 (0.110)**	**12.100 (1.750)**	0.754 (0.107)	10.500 (2.500)
0.9, 0.6, 0.7	**0.860 (0.070)**	**12.500 (1.360)**	0.753 (0.131)	10.800 (1.833)
0.7, 0.3, 0.9	**0.960 (0.060)**	**14.000 (1.340)**	0.825 (0.061)	11.100 (1.375)
0.7, 0.3, 0.7	**0.960 (0.060)**	**13.80 (1.460)**	0.821 (0.153)	11.600 (2.498)
0.7, 0.6, 0.9	**0.670 (0.150)**	**9.200 (1.980)**	0.641 (0.116)	10.300 (1.269)
0.7, 0.6, 0.7	**0.620 (0.100)**	**9.300 (1.480)**	0.590 (0.124)	11.100 (2.211)

The best results are highlighted in bold-face. The average ARI index and its standard deviation values are presented over 10 different data sets

Table 17 The performance of the SEFNACs and SEFNACn on small-size networks with quantitative features at the nodes after 50% noise features are inserted

p, q, α	SEFNACs		SEFNACn	
Setting p, q, α	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	0.998 (0.007)	5.000 (0.000)	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.3, 0.7	0.987 (0.031)	5.000 (0.000)	**0.999 (0.003)**	**5.000 (0.000)**
0.9, 0.6, 0.9	**0.959 (0.077)**	**4.900 (0.300)**	0.918 (0.081)	4.600 (0.490)
0.9, 0.6, 0.7	**0.978 (0.019)**	**5.000 (0.000)**	0.911 (0.081)	4.900 (0.700)
0.7, 0.3, 0.9	**0.988 (0.037)**	**5.000 (0.000)**	0.986 (0.042)	4.900 (0.300)
0.7, 0.3, 0.7	**1.000 (0.000)**	**5.000 (0.000)**	0.974 (0.058)	4.900 (0.300)
0.7, 0.6, 0.9	**0.765 (0.167)**	**4.700 (1.345)**	0.760 (0.105)	5.800 (2.182)
0.7, 0.6, 0.7	**0.720 (0.175)**	**4.800 (0.979)**	0.716 (0.146)	6.200 (1.077)

The best results are highlighted in bold-face. The average ARI index and its standard deviation values are presented over 10 different data sets

Table 18 The performance of the SEFNACs and SEFNACn on medium-size networks with quantitative features at the nodes with 50% noise features inserted

p, q, α	SEFNACs		SEFNACn	
Setting p, q, α	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	**0.999 (0.001)**	**15.000 (0.000)**	0.976 (0.039)	14.200 (1.077)
0.9, 0.3, 0.7	**0.999 (0.001)**	**15.000 (0.000)**	0.815 (0.245)	11.700 (3.662)
0.9, 0.6, 0.9	**0.866 (0.111)**	**12.400 (1.562)**	0.695 (0.081)	9.600 (1.562)
0.9, 0.6, 0.7	**0.848 (0.101)**	**11.900 (1.813)**	0.788 (0.112)	10.800 (1.600)
0.7, 0.3, 0.9	**0.973 (0.027)**	**14.100 (0.943)**	0.788 (0.093)	10.800 (1.600)
0.7, 0.3, 0.7	**0.932 (0.114)**	**13.500 (2.012)**	0.817 (0.138)	11.400 (2.417)
0.7, 0.6, 0.9	**0.674 (0.109)**	**9.500 (0.921)**	0.632 (0.117)	10.800 (1.249)
0.7, 0.6, 0.7	0.590 (0.104)	8.900 (1.220)	**0.621 (0.107)**	**12.000 (2.366)**

The best results are highlighted in bold-face. The average ARI index and its standard deviation over 10 different data sets values are presented

Table 19 The performance of the SEFNACs and SEFNACn on small-size networks with combined quantitative and categorical features at the nodes

$p, q, \alpha \vert \epsilon$	SEFNACs		SEFNACn	
Setting $p, q, \alpha \vert \epsilon$	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	0.996 (0.011)	5.000 (0.000)	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.3, 0.7	0.982 (0.025)	5.000 (0.000)	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.6, 0.9	**0.914 (0.093)**	**4.600 (0.490)**	0.841 (0.084)	4.800 (0.748)
0.9, 0.6, 0.7	**0.865 (0.139)**	**4.800 (0.600)**	0.839 (0.115)	5.300 (1.187)
0.7, 0.3, 0.9	**0.993 (0.016)**	**5.000 (0.000)**	0.961 (0.064)	4.800 (0.400)
0.7, 0.3, 0.7	0.943 (0.105)	4.900 (0.300)	**0.952 (0.061)**	**4.800 (0.400)**
0.7, 0.6, 0.9	0.736 (0.197)	3.800 (0.872)	**0.745 (0.111)**	**6.000 (1.000)**
0.7, 0.6, 0.7	**0.665 (0.143)**	**4.300 (1.100)**	0.587 (0.115)	7.300 (1.616)

The best results are highlighted in bold-face. The average ARI index and its standard deviation values are presented over 10 different data sets

Regarding the obtained results in Table 19 we see that each of the SEFNACs and SEFNACn wins at four different settings. This behavior shows that each of these two models can be more effective, than the other one, in specific circumstances.

Table 20 shows the results of our methods over the combination of quantitative and categorical features for medium-size networks. The SEFNACs are the winner. The results by the SEFNACn are moderately acceptable.

The following two tables present the results by our methods on small-size and medium-size networks with the combination of quantitative and categorical features at its nodes when 50% noise features are added.

Both SEFNACs and SEFNACn show rather robust results against the noise according to Tables 21 and 22. In particular, the level of cluster recovery by SEFNACn, ARI=0.950, at $(p, q, \alpha) = (0.7, 0.3, 0.7)$, is indeed impressive (Table 21).

On medium-size networks with added noise, the SEFNACs dominate rather clearly, although both methods lead to poor results in the last two rows. Also, one can see that the SEFNACn is quite sensitive to raising the q, the probability of between-community links, from 0.3 to 0 .6: its performance lowers then rather significantly.

All over, noteworthy, to point out that SEFNACs are superior over SEFNACn on most of the settings, both on small-size and medium-size networks. This, probably, can be associated with the mechanism for link data generation in our experiments: the probabilities p and q are constant across the entire data.

Table 20 The performance of the SEFNACs and SEFNACn on medium-size networks with the combination of quantitative and categorical features at the nodes

p, q, α	the SEFNACs		SEFNACn	
Setting $p, q, \alpha \vert \epsilon$	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	**1.000 (0.000)**	**15.000 (0.000)**	0.942 (0.077)	13.200 (1.939)
0.9, 0.3, 0.7	**1.000 (0.000)**	**15.000 (0.000)**	0.900 (0.200)	12.900 (3.081)
0.9, 0.6, 0.9	**0.948 (0.076)**	**14.000 (1.265)**	0.718 (0.105)	10.700 (1.487)
0.9, 0.6, 0.7	**0.842 (0.085)**	**12.100 (1.221)**	0.680 (0.093)	11.200 (1.778)
0.7, 0.3, 0.9	**0.994 (0.014)**	**14.900 (0.300)**	0.885 (0.098)	12.600 (1.562)
0.7, 0.3, 0.7	**0.994 (0.008)**	**14.800 (0.400)**	0.819 (0.123)	12.000 (1.897)
0.7, 0.6, 0.9	**0.565 (0.137)**	**7.800 (1.778)**	0.545 (0.096)	12.100 (1.700)
0.7, 0.6, 0.7	0.391 (0.093)	7.100 (1.513)	**0.431 (0.048)**	**13.200 (1.166)**

The best results are highlighted in bold-face. The average ARI index and its standard deviation over 10 different data sets values are presented

Table 21 The performance of the SEFNACs and SEFNACn on small-size networks with combined quantitative and categorical features at the nodes with 50% noise features replicated

$p, q, \alpha \vert \epsilon$	SEFNACs		SEFNACn	
Setting p, q, α	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	0.992 (0.012)	5.000 (0.000)	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.3, 0.7	0.990 (0.019)	5.000 (0.000)	**1.000 (0.000)**	**5.000 (0.000)**
0.9, 0.6, 0.9	**0.883 (0.099)**	4.500 (0.670)	0.848 (0.104)	**5.000 (0.632)**
0.9, 0.6, 0.7	**0.876 (0.142)**	**4.800 (0.399)**	0.821 (0.131)	5.500 (0.922)
0.7, 0.3, 0.9	**0.990 (0.014)**	**5.000 (0.000)**	0.970 (0.060)	4.900 (0.300)
0.7, 0.3, 0.7	0.726 (0.147)	4.200 (0.871)	**0.950 (0.062)**	**4.900 (0.300)**
0.7, 0.6, 0.9	0.726 (0.147)	4.200 (0.871)	**0.731 (0.119)**	6.300 (1.100)
0.7, 0.6, 0.7	**0.569 (0.144)**	**3.900 (0.538)**	0.560 (0.066)	7.800 (1.166)

The best results are highlighted in bold-face. The average ARI index and its standard deviation over 10 different data sets values are presented

Table 22 The performance of SEFNACs and SEFNACn on medium-size networks with combined quantitative and categorical features at the nodes when 50% noise features are involved

$p, q, \alpha \vert \epsilon$	SEFNACs		SEFNACn	
Setting p, q, α	ARI mean (std)	K mean (std)	ARI mean (std)	K mean (std)
0.9, 0.3, 0.9	**1.000 (0.001)**	**15.000 (0.000)**	0.948 (0.062)	13.200 (1.720)
0.9, 0.3, 0.7	**0.998 (0.002)**	**15.000 (0.000)**	0.880 (0.198)	12.300 (3.257)
0.9, 0.6, 0.9	**0.934 (0.103)**	**13.700 (1.676)**	0.746 (0.092)	10.900 (1.578)
0.9, 0.6, 0.7	**0.810 (0.085)**	**11.800 (1.470)**	0.659 (0.067)	10.500 (1.628)
0.7, 0.3, 0.9	**0.996 (0.006)**	**14.900 (0.300)**	0.848 (0.107)	11.500 (2.062)
0.7, 0.3, 0.7	**0.964 (0.066)**	**14.300 (1.187)**	0.845 (0.064)	12.000 (1.342)
0.7, 0.6, 0.9	0.547 (0.142)	8.100 (1.700)	0.553 (0.059)	11.900 (1.375)
0.7, 0.6, 0.7	**0.415 (0.078)**	**7.400 (0.663)**	0.408 (0.068)	12.900 (1.513)

The best results are highlighted in bold-face. The average ARI index and its standard deviation over 10 different data sets are presented

5.3 Computational complexity of SEFNAC algorithms

SEFNAC method is compute-intensive: it computes and compares values of criteria in Eqs. (24) and (25) while finding a node to be added to a current community. In the current implementation, criterion (25) is computed in a vectorized form, whereas criterion (24) requires a nested "for" loop, which takes a longer time. Let us denote the average time for computing the value of criterion (25) by tn, and the value of criterion (24), by ts. Then the total time for execution of SEFNACn (or, SEFNACs) is proportional to $tn * N^2$ (or, $ts * N^2$). Indeed, to find a community, SEFNAC adds a number of nodes proportional to N, and selecting a node to be added at a step, requires a number of tries proportional to N too. We do not take into account the number of communities found, because it is always finite in our computations.

To check whether these timings go in line with timings by the other algorithms, we took the common ground, synthetic networks with categorical features only, and ran all the algorithms at both small-sized data sets and medium-sized data sets. The computing time should not much depend on the parameter setting, so we selected two out of our standard 8 settings, $(p, q, \epsilon) = (0.9, 0.3, 0.9)$, at which the community structure is maximally sharp, and $(p, q, \epsilon) = (0.7, 0.6, 0.7)$, at which the community structure is maximally blurred.

Although the computation speed depends on the computing system, so that only relative comparisons can be meaningful, the times reported in the following Table 23 have been achieved at a desktop computer Intel(R) (Core(TM) i9-9900K CPU /@ 3.60GHz, RAM: 64 GB, HD: 1TB SSD) under Ubuntu 18.0 Operation System.

As one can see, the algorithms can be divided in two leagues, fast and slow. The fast league comprises CESNA, EVA, and SEFNACn, and slow league comprises SIAN and SEFNACs, although some may claim that, in fact, SEFNACs is much faster than SIAN. Anyway, an important fact is that computing time by SEFNAC algorithms goes in line with the flock.

6 Conclusion and future work

This paper proposes two similar methods, SEFNACs and SEFNACn, for community detection at feature-rich networks using the conventional data recovery approach. The methods differ in the assumptions of the network data entries' summability across the link table, yes or no, respectively. In this way, we distinguish between cases where the similarity data scales are the same for all the network nodes and cases at which each node collects its linkage data independently. The methods are similar in that both find clusters one-by-one and apply the same strategy for finding an individual cluster, this time by adding entities to the cluster one at a time. The summability and nonsummability assumptions lead to differing results. The nonsummable version, SEFNACn, is much faster than the summable SEFNACs. In contrast to our intuition, the nonsummable version leads to the best results on two of the six real-world data sets and second-best results on three out of four remaining ones. It brings forth the best results on six out of eight small-sized synthetic data sets (and second-best results on the two remaining settings) at the only case at which all the five algorithms under comparison can meet—networks with categorical features at the nodes. SEFNACs dominate at the medium-sized feature-rich networks.

There are several properties distinguishing our methods from many others.

Desirable properties: (a) methods can work with mixed scale features, both categorical or quantitative; (b) methods apply to any link data formats that can be expressed as entity-to-entity similarity matrices; (c) the number of clusters/communities found is determined automatically; (d) contributions of individual clusters to the combined data scatter are computed.

Less desirable properties: (e) results depend on the data standardization, both network link data, and feature data; (f) computations are slow (of the order of N^2, in the worst-case scenario); (g) there is no advice regarding the constants balancing the relative contributions of network and features.

Nevertheless, our experiments show that our algorithms are competitive against state-of-the-art algorithms on small-size and medium-size data. The algorithms are relatively robust against noise and blurred structures. For instance, the probability of inter-community links can be as high as 0.6, meaning that the proportion of inter-community edges may be comparable with or even greater than the probability of within-community edges.

Our empirically found best data standardization options involve z-scoring of the feature data and uniform shift transformation of the network data in almost all settings. The

Table 23 The execution time of various methods on synthetic networks with categorical features at the nodes; for SEFNAC, the selected data preprocessing options apply

p, q, ϵ	CESNA		SIAN		EVA		SEFNACs		SEFNACn	
	Small	Medium	Small	Medium	Small	Medium	Small	Medium	Small	Medium
0.9, 0.3, 0.9	0.442	38.265	95.949	856.785	0.679	19.554	4.647	492.006	1.282	47.162
0.7, 0.6, 0.7	0.699	83.961	335.198	2674.541	0.831	54.790	3.652	476.251	1.268	65.635

The average of time at 10 different data sets of the same setting is reported, in seconds

uniform shift subtracts a constant threshold from the link values. In contrast, the popular modularity transformation subtracts random noise, which may differ depending on the number of links at different nodes. This supports the view (Mirkin 2012) that at flat network data, the subtracted value should be flat/constant, too. The reason why Z-scoring improves our results seems more obscure. Perhaps, an explanation might sound like this. Recall that the Z-scoring leads to different feature ranges, in contrast to the Range standardization. This may work at our method, since it starts from the most anomalous clusters.

The properties of the methods mentioned above determine the main directions of future work. First of all, we should raise the computational power of the method to be applied to larger data sets. We think that a version resembling the K-Means method can be developed to define the cluster center and distances from that to cluster elements according to our clustering criterion (5) and, in this way, take into account both data sources. Such a development could lead us to an algorithm capable of handling dozens of thousand nodes rather than thousands, the current version's capability. Then we could think of a further acceleration of the computations.

We are going to look for a practically interesting real-world application. Having such an application may help us improve the speed of computations.

Another direction for research would be in investigating the balancing weights between the network links and feature values to adjust the trade-off coefficients automatically.

Finally, reformulating our proposed methods in a theory-driven form seems another promising direction of future work.

Acknowledgements This article was prepared within the framework of the HSE University Basic Research Program. The authors are indebted to the anonymous referees for their invaluable comments taken into account in the final draft.

References

Akoglu L, Tong H, Meeder B, Faloutsos C (2012) PICS: parameter-free identification of cohesive subgroups in large attributed graphs. In: Proceedings of the 12th SIAM international conference on data mining, pp 439–450, SDM

Amorim RC, Mirkin B (2012) Minkowski metric, feature weighting and anomalous cluster initializing in K-Means clustering. Pattern Recognit 45(3):1061–1075

Bojchevski A, Günnemann S (2018) Bayesian robust attributed graph clustering: joint learning of partial anomalies and group structure. In: Proceedings of thirty-second AAAI conference on artificial intelligence, pp 2738–2745, https://aaai.org/Library/AAAI/aaai18contents.php, Accessed 13 Oct 2020

Binkiewicz N, Vogelstein JT, Rohe K (2017) Covariate-assisted spectral clustering. Biometrika 104(2):361–377

Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 10:P10008

Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge

Cao J, Wang H, Jin D, Dang J (2019) Combination of links and node contents for community discovery using a graph regularization approach. Future Gener Comput Syst 91:361–370

Cavallari S, Zheng VW, Cai H, Chang KCC, Cambria E (2017) Learning community embedding with community detection and node embedding on graphs. In: Proceedings of the 2017 ACM conference on information and knowledge management. ACM, pp 377–386

Citraro S, Rossetti G (2020) Identifying and exploiting homogeneous communities in labeled networks. Appl Netw Sci 5(1):1–20

Chang S, Han W, Tang J, Qi GJ, Aggarwal CC, Huang TS (2015) Heterogeneous network embedding via deep architectures. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp 119–128

Chiang MMT, Mirkin B (2010) Intelligent choice of the number of clusters in k-means clustering: an experimental study with different cluster spreads. J Classif 27(1):3–40

Chunaev P (2020) Community detection in node-attributed social networks: a survey. Comput Sci Rev 37:100286

Combe D, Largeron C, Géry M, Egyed-Zsigmond E (2015) I-louvain: an attributed graph clustering method. In: Fromont E, De Bie T, van Leeuwen M (eds) Advances in intelligent data analysis XIV. Springer International Publishing, Cham, pp 181–192

Cross RL, Parker A (2004) The hidden power of social networks: understanding how work really gets done in organizations. Harvard Business Press, Boston

Cover TM, Thomas JA (2012) Elements of information theory. John Wiley and Sons, New York

De Nooy W, Mrvar A, Batagelj V (2004) Exploratory social network analysis with Pajek. Cambridge University Press, Cambridge

Dang TA, Viennet E (2012) Community detection based on structural and attribute similarities. In: International conference on digital society (ICDS), pp 7–12

Doreian P, Batagelj V, Ferligoj A (2020) Advances in network clustering and blockmodeling. John Wiley and Sons, New York

Goldenberg A, Zheng AX, Fienberg SE, Airoldi EM (2010) A survey of statistical network models. Now Publishers Inc., Norwell

GitHub Repository, Giulio Rossetti, Italian National Research Council https://github.com/GiulioRossetti/EVA

Hoffman M, Steinley D, Gates KM, Prinstein MJ, Brusco MJ (2018) Detecting clusters/communities in social networks. Multivar Behav Res 53(1):57–73

Holland PW, Laskey KB, Leinhardt S (1983) Stochastic blockmodels: first steps. Soc Networks 5(2):109–137

Hu Y, Li M, Zhang P, Fan Y, Di Z (2008) Community detection by signaling on complex networks. Phys Rev E 78(1):16115

Hubert L, Arabie P (1985) Comparing partitions. J Classif 2(1):193–218

Interdonato R, Atzmueller M, Gaito S, Kanawati R, Largeron C, Sala A (2019) Feature-rich networks: going beyond complex network topologies. Appl Network Sci 4:1–13

Javed MA, Younis MS, Latif S, Qadir J, Baig A (2018) Community detection in networks: a multidisciplinary review. J Network Comput Appl 108:87–111

Jia C, Li Y, Carson MB, Wang X, Yu J (2017) Node attribute-enhanced community detection in complex networks. Sci Rep 7(1):2626

Jin H, Yu W, Li S (2018) A clustering algorithm for determining community structure in complex networks. Phys A Stat Mech Appl 492:980–993

Kovaleva EV, Mirkin B (2015) Bisecting K-means and 1D projection divisive clustering: a unified framework and experimental comparison. J Classif 32(3):414–442

Larremore DB, Clauset A, Buckee CO (2013) A network approach to analyzing highly recombinant malaria parasite genes. PLoS Comput Biol 9(10):e1003268

Lazega E (2001) The collegial phenomenon: the social mechanisms of cooperation among peers in a corporate law partnership. Oxford University Press, Oxford

Leskovec J, Sosič R (2016) SNAP: a general-purpose network analysis and graph-mining library, ACM transactions on intelligent systems and technology (TIST), vol 8-1, p 1, ACM, CESNA on Github: https://github.com/snap-stanford/snap/tree/master/examples/cesna

Mirkin B (1987) Additive clustering and qualitative factor analysis methods for similarity matrices. J Classif 4:7–31

Mirkin B (1990) A sequential fitting procedure for linear data analysis models. J Classif 7(2):167–195

Mirkin B (2008) The iterative extraction approach to clustering. In: Gorban A (ed) Principal manifolds for data visualization and dimension reduction. Springer, Heidelberg, pp 151–177

Mirkin B, Nascimento S (2012) Additive spectral method for fuzzy cluster analysis of similarity data including community structure and affinity matrices. Inf Sci 183(1):16–34

Mirkin B (2012) Clustering: a data recovery approach, CRC Press, 1st Edition, 2005; 2d Edition, 2012

Nascimento S, Casca S, Mirkin B (2015) A seed expanding cluster algorithm for deriving upwelling areas on sea surface temperature images. Comput Geosci 85:74–85

Nature Communications, Mark J. Newman, "W DC University", https://www.nature.com/articles/ncomms11863

Newman ME (2006) Modularity and community structure in networks. Proc Natl Acad Sci 103(23):8577–8582

Newman ME, Clauset A (2016) Structure and inference in annotated networks. Nat Commun 7:11863

Neville J, Adler M, Jensen D (2003) Clustering relational data using attribute and link information. In: Proceedings of the text mining and link analysis workshop, 18th international joint conference on artificial intelligence (pp 9–15). San Francisco, CA, Morgan Kaufmann Publishers

Ng A (2011) Sparse autoencoder, CS294A lecture notes 72, pp 1–19

Nowicki K, Snijders TAB (2001) Estimation and prediction for stochastic blockstructures. J Am Stat Assoc 96(455):1077–1087

Peel L, Larremore DB, Clauset A (2017) The ground truth about metadata and community detection in networks. Sci Adv 3(5):e1602548

Robin G, Klopp O, Josse J, Moulines É, Tibshirani R (2019) Main effects and interactions in mixed and incomplete data frames. J Am Stat Assoc (Accepted), pp 1–31

Sánchez PI, Müller E, Korn UL, Böhm K, Kappes A, Hartmann T, Wagner D (2015) Efficient algorithms for a robust modularity-driven clustering of attributed graphs. In: Proceedings of the 2015 SIAM International conference on data mining, pp 100–108

Shalileh S, Mirkin B (2020) A data recovery method for community detection in feature-rich networks. In: Proceedings of the 2020 IEEE/ACM international conference on advances in social networks analysis and mining, pp 99–104

Shi J, Malik J (2000) Normalized cuts and image segmentation. IEEE Trans Pattern Anal Mach Intell 22(8):888–905

Stanley N, Bonacci T, Kwitt R, Niethammer M, Mucha PJ (2019) Stochastic block models with multiple continuous attributes. Appl Network Sci 4(1):1–22

Snijders T. The Siena webpage. https://www.stats.ox.ac.uk/~snijders/siena/Lazega_lawyers_data.htm

Strehl A, Ghosh J (2002) Cluster ensembles—a knowledge reuse framework for combining multiple partitions. J Mach Learn Res, pp 583–617

Sun H, He F, Huang J, Sun Y, Li Y, Wang C, He L, Sun Z, Jia X (2020) Network embedding for community detection in attributed networks. ACM Trans Knowl Discov Data (TKDD) 14(3):1–25

Vichi M (2008) Fitting semiparametric clustering models to dissimilarity data. Adv Data Anal Classif 2(2):121–161

Wang D, Zhao Y (2019) Network community detection from the perspective of time series. Phys A Stat Mech Appl 522:205–214

Wang X, Jin D, Cao X, Yang L, Zhang W (2016) Semantic community identification in large attribute networks. In: Proceedings of the thirtieth AAAI conference on artificial intelligence, AAAI'16, pp 265–271. AAAI Press

Xu Z, Ke Y, Wang Y, Cheng H, Cheng J (2012) A model-based approach to attributed graph clustering. In: Proceedings of the 2012 ACM SIGMOD international conference on management of data, pp 505–516. ACM

Ye W, Zhou L, Sun X, Plant C, Böhm C (2017) Attributed graph clustering with unimodal normalized cut. In: Ceci M, Hollmén J, Todorovski L, Vens C, Džeroski S (eds) Machine learning and knowledge discovery in databases. Springer International Publishing, Cham, pp 601–616

Yang J, McAuley J, Leskovec J (2013) Community detection in networks with node attributes. In: 2013 IEEE 13th international conference on data mining (pp 1151–1156). IEEE, arXiv:1401.7267, Accessed 22 Nov 2019

Zanghi H, Volant S, Ambroise C (2010) Clustering based on random graph model embedding vertex features. Pattern Recognit Lett 31(9):830–836

Zhang Y, Levina E, Zhu J (2015) Community detection in networks with node features. arXiv preprint arXiv:1509.01173