Data Min Knowl Disc (2012) 25:1–33
DOI 10.1007/s10618-011-0231-0

Community detection via heterogeneous interaction analysis

Lei Tang · Xufei Wang · Huan Liu

Received: 7 June 2010 / Accepted: 3 August 2011 / Published online: 21 August 2011

Abstract The pervasiveness of Web 2.0 and social networking sites has enabled people to interact with each other easily through various social media. For instance, popular sites like Del.icio.us, Flickr, and YouTube allow users to comment on shared content (bookmarks, photos, videos), and users can tag their favorite content. Users can also connect with one another, and subscribe to or become a fan or a follower of others. These diverse activities result in a *multi-dimensional network* among actors, forming group structures with group members sharing similar interests or affiliations. This work systematically addresses two challenges. First, it is challenging to effectively integrate interactions over multiple dimensions to discover hidden community structures shared by heterogeneous interactions. We show that representative community detection methods for single-dimensional networks can be presented in a unified view. Based on this unified view, we present and analyze four possible integration strategies to extend community detection from single-dimensional to multi-dimensional networks. In particular, we propose a novel integration scheme based on structural features. Another challenge is the evaluation of different methods without ground truth information about community membership. We employ a novel cross-dimension network validation (CDNV) procedure to compare the performance of different methods.

Responsible editor: Eamonn Keogh.

L. Tang (✉)
Yahoo! Labs Silicon Valley, Santa Clara, CA 95054, USA
e-mail: L.Tang@asu.edu

X. Wang · H. Liu
Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287, USA
e-mail: Xufei.Wang@asu.edu

H. Liu
e-mail: Huan.Liu@asu.edu

We use synthetic data to deepen our understanding, and real-world data to compare integration strategies as well as baseline methods in a large scale. We study further the computational time of different methods, normalization effect during integration, sensitivity to related parameters, and alternative community detection methods for integration.

Keywords Community detection · Heterogeneous interactions · Network integration · Multi-dimensional networks · Social media

1 Introduction

The recent boom of social media (e.g., Del.icio.us, Flickr, YouTube, Facebook, MySpace and Twitter) permits human interaction with unprecedented convenience. With widely-available large-scale networks in social media, social network analysis is gaining increasing attention from a variety of disciplines including computer science, physics, economics, epidemiology, business marketing, and behavioral science. One fundamental task is to find cohesive subgroups (a.k.a. communities) whose members interact more frequently with each other than with those outside the group (Wasserman and Faust 1994). The extracted communities can be utilized for further analysis such as visualization (Kang et al. 2007), viral marketing (Richardson and Domingos 2002), determining the causal factors of group formation (Backstrom et al. 2006), detecting group evolution (Palla et al. 2007) or stable clusters (Bansal et al. 2007), relational learning (Tang and Liu 2009a), and building ontology for semantic web (Mika 2007; Specia and Motta 2007; Lizorkin et al. 2009).

A plethora of approaches have been proposed to address community detection with network data. However, most existing work focuses on only one dimension of interaction among people (i.e., a network comprised of interactions of a single type). In reality, people interact with each other in assorted forms of activities, leading to multiple networks among the same set of actors, or a *multi-dimensional network*[1] with each dimension representing one type of interaction. In the physical world, people interact with others in a variety of ways, e.g., face-to-face, by email or by phone; The same is true in cyberspace as shown in Fig. 1. For instance, at popular photo and video sharing sites (Flickr and YouTube), a user can connect to his friends through email invitations or the provided "add as contacts" function. Users can also tag/comment on shared content such as photos and videos. A user on YouTube can upload a video to respond to a video posted by another user, and can also become a fan of another user by "subscribing" to the user's content. Networks can be constructed based on each form of activity. By combining them together, we obtain a multi-dimensional network representing the richness of user interaction. More generally, people

[1] Some researchers also use the paraphrase *multi-relational network*. In social science, multi-relational network tends to refer to the case that multiple different relations exist between two actors. While in computer science domain, multi-relational network tends to refer a network with heterogeneous entities interacting with each other, which actually corresponds to a *multi-mode* (or *multi-modal*) *network* (Tang et al. 2008). Here, we use *multi-dimensional network* to emphasize that actors are involved in disparate interactions. We also notice that some researchers refer to this kind of network as *multiplex network* (Mucha et al. 2010).

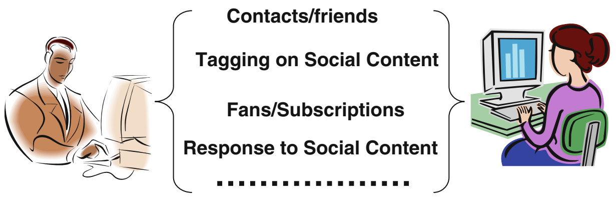

Fig. 1 Multi-dimensional network

can be active at multiple different social networking sites. It is common for one user to be registered on several social networking sites at the same time, e.g., Facebook, Twitter, BlogSpot, YouTube, and Del.icio.us. In such cases, a multi-dimensional network can be constructed with each dimension representing user interaction at each site.

For a multi-dimensional network with heterogeneous interactions, one type of interaction might be insufficient to determine group membership accurately. In social media, a certain type of interaction can be incomplete due to users' privacy concern. The interactions can also be noisy since it is much easier to connect with another user online than in the physical world. Indeed, some online users have thousands of online friends whereas this could hardly be true in reality. For instance, one user in Flickr has more than 19,000 friends. For this kind of user, it is really fuzzy to mine the community he/she is involved in using the friendship network alone. On the other hand, many users in the network might have only one or two friends. With these noisy and highly imbalanced interactions, relying on one type of interaction alone might miss the true user community structure.

A multi-dimensional network may exhibit correlated or disparate community structure in each dimension. But in this article, we are concentrating on finding a shared community structure underneath different interactions. One motivating application from Yahoo! Inc. is that online users engage in various types of interactions through different Yahoo! properties. The interactions include friendship information from Yahoo! instant messenger (IM) buddylist, IM communications, email address book, email communications, and interactions at Yahoo pulse (a social networking platform similar to Twitter). One particular demand from the advertising team is to utilize these various types of interactions to help for targeting, and to locate user segment and communities that are correlated with specific topics. Once the community information is extracted, we may combine it with other behavioral features (say, page views, search queries) to determine user interests for serving appropriate ads. We notice that most users engage in only one or two interactions. In this particular application, it becomes a pressing need to integrate all kinds of interactions together to find the community structure among users.

Integrating assorted forms of interaction can compensate for incomplete information in each dimension, reduce noise, and obtain a more reliable community structure. Some users might be reluctant to add friends, but frequently engage in another activity such as uploading videos or commenting on other videos. The interactions at different dimensions all indicate user interests. Hence, one might infer a more accurate community structure by integrating disparate interactions. However, idiosyncratic personalities lead to varied local correlations between dimensions. Some people interact with group members consistently in one form of activity, but infrequently in another. It thus becomes a challenge to identify groups in multi-dimensional networks because we have to fuse the information from all dimensions for integrated analysis.

In this work, we first present representative approaches of community detection with a unified view. Based on this unified view, we discuss potential extensions of community detection in one-dimensional (1-D) networks to multi-dimensional (M-D) networks. We present four integration strategies in terms of network interactions, utility functions, structural features and community partitions, respectively. Their pros and cons are discussed in detail. Typically, a real-world network does not have full information about group membership. Hence, a novel CDNV procedure is proposed to compare the communities obtained from different approaches. We establish the veracity of this evaluation scheme based on synthetic data with known community structure, and then apply it to a real-world network data to systematically compare different integration strategies.

2 Community detection in 1-D networks

In this section, we review existing representative methods for community detection in 1-D networks, and then present a unified view of these methods, preparing their extension to multi-dimensional networks.

Let $G(V, E)$ denote a network with V the set of n vertices and E the m edges, and $A \in \{0, 1\}^{n\times n}$ denote the adjacency matrix (network interactions). The degree of node i is d_i. $A_{ij} = 1$ if there is an edge between nodes i and j. Unless specified explicitly, we assume the network is undirected. A community is defined as a group of actors with frequent interactions occurring between them. Community detection attempts to uncover the community membership of each actor. In particular, the problem is defined below:

Community Detection: Given a network $A \in \{0, 1\}^{n\times n}$ with n being the number of actors, and k the number of communities in the network, community detection aims to determine the community assignment of each actor. The community assignment is denoted as $H \in \{0, 1\}^{n\times k}$ with

$$H_{ij} = \begin{cases} 1, & \text{if actor } i \text{ belongs to community } j \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

In this work, we study the case each actor belongs to only one community. That is, $\sum_{j=1}^{k} H_{ij} = 1$. To resolve the community detection problem, various approaches have been developed including latent space models, block model approximation, spectral clustering and modularity maximization. Below, we briefly review these representative methods and show that they can be interpreted in a unified view.

2.1 Latent space models

A latent space model maps the nodes in a network into a low-dimensional Euclidean space such that the proximity between the nodes based on network connectivity are kept in the new space, then the nodes are clustered in the low-dimensional space using methods like k-means (Witten and Frank 2005). One representative approach is *multi-dimensional scaling* (MDS) (Borg and Groenen 2005). Typically, MDS requires the input of a proximity matrix $P \in \mathbb{R}^{n \times n}$, with each entry P_{ij} denoting the distance between a pair of nodes i and j in the network. For a network, a commonly used proximity measure is *geodesic distance* (Wasserman and Faust 1994), i.e., the length of the shortest path between two nodes. Let $S \in \mathbb{R}^{n \times \ell}$ denote the coordinates of nodes in the ℓ-dimensional space such that S are column orthogonal. It can be shown (Borg and Groenen 2005; Sarkar and Moore 2005) that

$$SS^T \approx -\frac{1}{2}\left(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T\right)(P \circ P)\left(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T\right) = \widetilde{P} \tag{2}$$

where I is the identity matrix, $\mathbf{1}$ an n-dimensional column vector with each entry being 1, and $\circ$ the element-wise matrix multiplication. It follows that S can be obtained via minimizing the discrepancy between $\widetilde{P}$ and SS^T as follows:

$$\min \|SS^T - \widetilde{P}\|_F^2 \tag{3}$$

Suppose V are the top ℓ eigenvectors of $\widetilde{P}$ with largest eigenvalues, Λ a diagonal matrix of top ℓ eigenvalues $\Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_\ell)$. The optimal S is $S = V\Lambda^{\frac{1}{2}}$. Note that this MDS corresponds to an eigenvector problem of matrix $\widetilde{P}$. Then classical k-means algorithm can be applied to find community partitions.

2.2 Block model approximation

Block model approximation is to approximate a given network by a block structure. The basic idea can be visualized in Fig. 2 where the left graph shows a network and

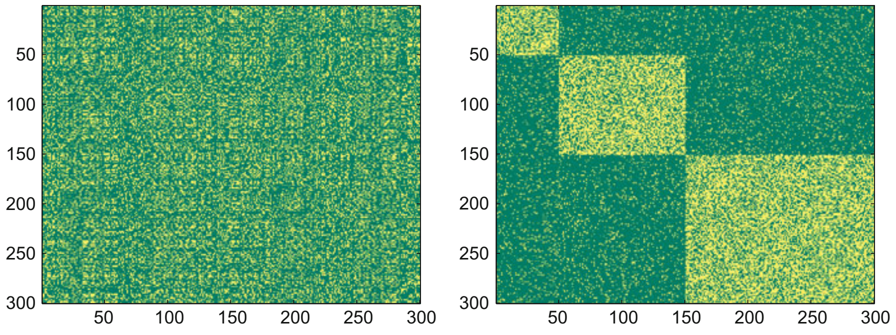

Fig. 2 Basic idea of block model approximation

the right one is the block structure after we reorder the index of actors according to their community membership. Each block represents one community. Therefore, we approximate the network interaction A as follows:

$$A \approx S\Sigma S^T \tag{4}$$

where $S \in \{0, 1\}^{n\times k}$ is the block indicator matrix, Σ the block (group) interaction density, and k the number of blocks. A natural objective is to minimize the following formula:

$$\min \|A - S\Sigma S^T\|_F^2 \tag{5}$$

The discreteness of S makes the problem NP-hard. We can relax S to be continuous but satisfy certain orthogonal constraints, i.e., $S^T S = I_k$, then the optimal S corresponds to the top k eigenvectors of A with maximum eigenvalues. Similar to the latent space model, k-means clustering can be applied to S to recover the community partition H.

2.3 Spectral clustering

Spectral clustering (Luxburg 2007) derives from the problem of graph partition.Graph partition aims to find out a partition such that the cut (the total number of edges between two disjoint sets of nodes) is minimized. Though this cut minimization can be solved efficiently, it often returns trivial and non-interesting singletons, i.e., a community consisting of only one node. Therefore, practitioners modify the objective function so that the group size of communities is considered. Two commonly used variants are *Ratio Cut* and *Normalized Cut*. Suppose we partition the nodes of a network into k non-overlapping communities $\pi = (C_1, C_2, \ldots, C_k)$, then

$$\text{Ratio Cut}(\pi) = \sum_{i=1}^{k} \frac{cut(C_i, \bar{C}_i)}{|C_i|} \tag{6}$$

$$\text{Normalized Cut}(\pi) = \sum_{i=1}^{k} \frac{cut(C_i, \bar{C}_i)}{vol(C_i)} \tag{7}$$

where $\bar{C}_i$ is the complement of C_i, and $vol(C_i) = \sum_{v\in C_i} d_v$. Both objectives attempt to minimize the number of edges between communities, yet avoid the bias of trivial-size communities like singletons. Both can be formulated as a min-trace problem like below

$$\min_{S\in\{0,1\}^{n\times k}} Tr(S^T \widetilde{L} S) \tag{8}$$

with $\widetilde{L}$ (graph Laplacian) defined as follows:

$$\widetilde{L} = \begin{cases} D - A & \text{(Ratio Cut)} \\ I - D^{-1/2} A D^{-1/2} & \text{(Normalized Cut)} \end{cases} \tag{9}$$

Akin to block model approximation, we solve the following spectral clustering problem based on a relaxation to S.

$$\min_S Tr(S^T \widetilde{L} S) \quad s.t.\ S^T S = I_k \tag{10}$$

Then, S corresponds to the top eigenvectors of $\widetilde{L}$ with smallest eigenvalues.

2.4 Modularity maximization

Modularity (Newman 2006b) is proposed specifically to measure the strength of a community partition for real-world networks by taking into account the degree distribution of nodes. Given a random network with n nodes and m edges, the expected number of edges between node i and j is $d_i d_j / 2m$ where d_i and d_j are the degrees of node i and j, respectively. So $A_{ij} - d_i d_j / 2m$ measures how far the network interaction between nodes i and j (A_{ij}) deviates from the expected random connections. Given a group of nodes C, the strength of community effect is defined as

$$\sum_{i \in C, j \in C} A_{ij} - d_i d_j / 2m.$$

If a network is partitioned into multiple groups, the overall community effect can be summed up as follows:

$$\sum_C \sum_{i \in C, j \in C} A_{ij} - d_i d_j / 2m.$$

Modularity is defined as

$$Q = \frac{1}{2m} \sum_C \sum_{i \in C, j \in C} A_{ij} - d_i d_j / 2m. \tag{11}$$

where the coefficient $1/2m$ is introduced to normalize the value between -1 and 1. Modularity calibrates the quality of community partitions thus can be used as an objective measure to optimize.

Let $B = A - \frac{\mathbf{d}\mathbf{d}^T}{2m}$, $\mathbf{s}_C \in \{0, 1\}^n$ be the community indicator of group C, and S the community indicator matrix, it follows that

$$Q = \frac{1}{2m} \sum_C \mathbf{s}_C^T B \mathbf{s}_C = \frac{1}{2m} Tr(S^T B S) = Tr(S^T \tilde{B} S) \tag{12}$$

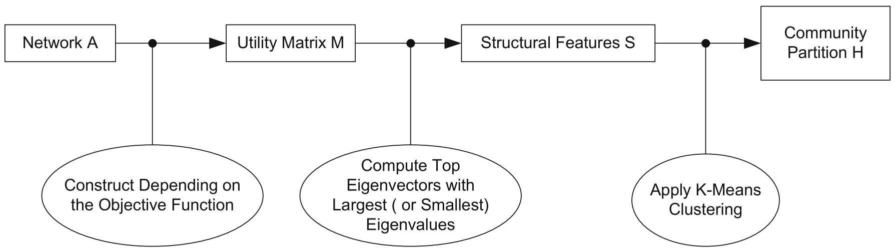
Network A
Utility Matrix M
Structural Features S
Community Partition H
Construct Depending on the Objective Function
Compute Top Eigenvectors with Largest (or Smallest) Eigenvalues
Apply K-Means Clustering

Fig. 3 A unified view of representative community detection methods

where

$$\tilde{B} = \frac{1}{2m}B = \frac{A}{2m} - \frac{\mathbf{d}\mathbf{d}^T}{(2m)^2}. \tag{13}$$

With a spectral relaxation to allow S to be continuous, the optimal S can be computed as the top-k eigenvectors of matrix $\widetilde{B}$ (Newman 2006a) with maximum eigenvalues.

2.5 A unified view

In the previous subsections, we briefly present four representative community detection methods: latent space models, block model approximation, spectral clustering and modularity maximization. Interestingly, all these methods can be unified in a process as in Fig. 3. The process is composed of four components with three intermediate steps. Given a network, a utility matrix is constructed. Depending on the objective function, different utility matrices can be constructed.

$$\text{Utility Matrix } M = \begin{cases} \widetilde{P} \text{ in Eq. 2} & \text{(latent space models)} \\ A \text{ in Eq. 4} & \text{(block model approximation)} \\ \widetilde{L} \text{ in Eq. 9} & \text{(spectral clustering)} \\ \widetilde{B} \text{ in Eq. 13} & \text{(modularity maximization)} \end{cases} \tag{14}$$

After obtaining the utility matrix, we obtain the *structural features* S via the top eigenvectors with largest (or smallest subject to formulation) eigenvalues. The selected eigenvectors capture the prominent interaction patterns, representing approximate community partitions. This step can also be considered as a de-noising process since we only keep those top eigenvectors that are indicative of community structures. To recover the discrete partition H, a k-means clustering algorithm is applied. Note that all the aforementioned approaches differ subtly by constructing different utility matrices.

2.5.1 Caveat

In this unified framework, we cover only some popular community detection methods. It is by no means to be complete in this focused article. For example, there are many probabilistic models to find community structure in graphs (Yu et al. 2005;

Airodi et al. 2008) and information theoretic clustering based on compressive coding (Rosvall and Bergstrom 2008). In order to optimize with respect to the utility function, spectral relaxation, as we introduced in previous subsections, has been adopted by many practitioners. But it is not the only option. Other types of optimization techniques include greedy method (Clauset et al. 2004), exhaustive search by simulated annealing (Guimera and Amaral 2005), multi-level method (Abou-Rjeili and Karypis 2006) and Markov Chain Monte Carlo method (MCMC). Interesting readers may refer to Fortunato (2010) for a comprehensive treatment. Comparison of some community detection methods on synthetic benchmark and real-world large scale networks can also be found at Lancichinetti and Fortunato (2009) and Leskovec et al. (2010). The unified framework present here will serve as a vehicle to derive different strategies for integration.

2.5.2 Time complexity

The community detection methods presented above, except the latent space model, are normally applicable to most medium-size networks (say, 100,000 nodes). The latent space model requires an input of a proximity matrix of the geodesic distance of any pair of nodes, which costs $O(n^3)$ to compute the pairwise shortest path distances. Moreover, the utility matrix of the latent space model is neither sparse nor structured, leading to $O(n^3)$ to compute its eigenvectors. This high computational cost hinders its application to real-world large-scale networks.

On the contrary, the other methods, block model approximation, spectral clustering, and modularity maximization, construct a sparse or structured (a sparse matrix plus low rank update) utility matrix, whose computational cost is almost negligible.[2] Asymptotically, the cost to construct a utility matrix is

$$T_{utility} = O(m). \tag{15}$$

Implicitly Restarted Lanczos method (IRLM) can be applied to compute the top eigenvectors efficiently (Demmel et al. 2000; White and Smyth 2005). Let ℓ denote the number of structural features to extract. If one makes the conservative assumption that $O(\ell)$ extra Lanczos steps be involved, IRLM has the worst time complexity of

$$T_{eig} = O(h(m\ell + n\ell^2 + \ell^3)) \tag{16}$$

where h, m and n are the number of iterations, the number of edges and nodes in the network, respectively. Typically, $m \sim O(n)$ in a social network with power law distribution (Tang and Liu 2009a) and $\ell \ll n$. In practice, the computation tends to be linear with respect to n if ℓ is small. The post-processing step to extract community partition relies on k-means clustering, which has time complexity

$$T_{kmeans} = O(nk\ell e) \tag{17}$$

[2] The utility matrix of modularity maximization is dense but structured, thus it is rarely computed out. Its structure is exploited directly for eigenvector computation (Newman 2006a; Tang et al. 2009a).

where ℓ is the number of structural features, e is the number of iterations.

In summary, the representative community detection methods can be unified in the same process. The only difference is how to construct the utility matrix. This also affects the time complexity of different methods. Block model approximation, spectral clustering, and modularity maximization share similar time complexity, which can be solved efficiently. With this unified view, we can systematically study different strategies to handle multi-dimensional networks.

3 Community detection in M-D networks

In the previous section, we reviewed various methods of community detection in 1-D networks and presented a unified view. Here, we systematically study possible strategies to extend community detection from 1-D networks to M-D networks. Before we proceed, we state the problem of M-D network community detection first.A d-dimensional network is represented as

$$\mathcal{A} = \{A^{(1)}, A^{(2)}, \ldots, A^{(d)}\}$$

with $A^{(i)}$ represents the interaction among actors in the ith dimension satisfying

$$A^{(i)} \in \mathbb{R}_{+}^{n \times n}, \quad A^{(i)} = (A^{(i)})^T, \quad i = 1, 2, \ldots, d$$

where n is the total number of actors involved in the network. Here, we concentrate on symmetric networks.[3] In a multi-dimensional network, the interactions of actors are represented in various forms. In certain scenarios, a latent community structure exists among actors, which explains these interactions. The goal of this work is to *infer the shared latent community structure among the actors given a multi-dimensional network*. In particular, we attempt to find out a community assignment such that a utility measure (e.g., block model approximation error, modularity) is optimized for each dimension.

In order to find out the shared community structure across multiple network dimensions, we have to integrate the information from all dimensions. Since four components (network, utility matrix, structural features and partition) are involved throughout the community detection process (Fig. 3), we can conduct the integration in terms of each component as in Fig. 4. In particular, we have *Network Integration*, *Utility Integration*, *Feature Integration*, and *Partition Integration*. Below, we delineate each type of integration strategy in detail. We use modularity maximization as an example to go through all the different strategies.[4] The derivation of other variants of community detection methods (such as block models and spectral clustering) in multi-dimensional networks following the unified view should be straightforward.

[3] Directed networks can be converted into undirected networks through certain operations as shown later.

[4] A preliminary work based on modularity maximization is published in Tang et al. (2009a). This manuscript, significantly different the previous conference version, presents a general framework to interpret community detection in multi-dimensional networks.

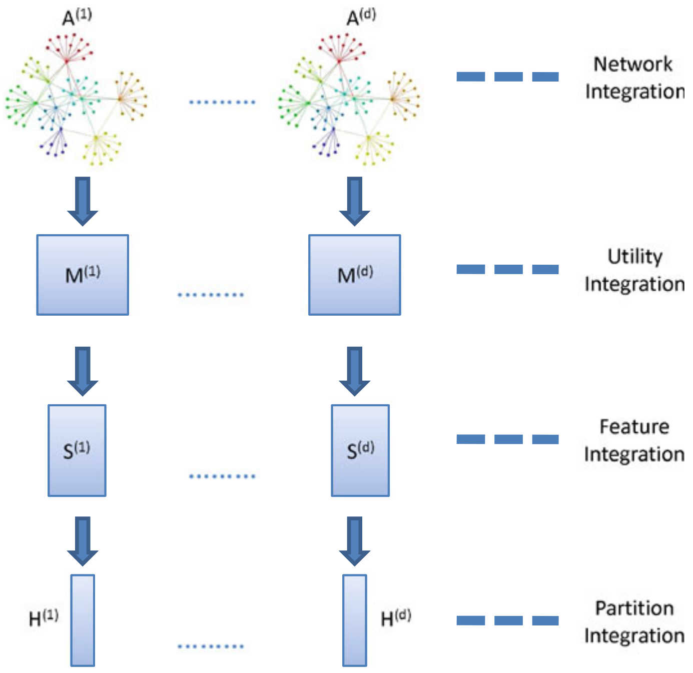

Fig. 4 Potential multi-dimensional integration strategies

3.1 Network integration

A simple strategy to handle a multi-dimensional network is to treat it as single-dimensional. One straightforward approach is to calculate the average interaction network among social actors:

$$\bar{A} = \frac{1}{d} \sum_{i=1}^{d} A^{(i)} \tag{18}$$

Correspondingly,

$$\bar{m} = \frac{1}{d} \sum_{i=1}^{d} m^{(i)}, \quad \bar{\mathbf{d}} = \frac{1}{d} \sum_{i=1}^{d} \mathbf{d}^{(i)} \tag{19}$$

With $\bar{A}$, this boils down to classical community detection in a single-dimensional network. Based on the average network, we can follow the community detection process as stated in the unified view. Take modularity maximization as an example. We can maximize the modularity as follows:

$$\max_{S} \frac{1}{2\bar{m}} Tr\left(S^T \left[\bar{A} - \frac{\bar{\mathbf{d}}\bar{\mathbf{d}}^T}{2\bar{m}} \right] S \right) \tag{20}$$

3.2 Utility integration

Another variant for integration is to combine utility matrices instead of networks. We can obtain an average utility matrix as follows:

$$\bar{M} = \frac{1}{d}\sum_{i=1}^{d} M^{(i)}$$

where $M^{(i)}$ denotes the utility matrix constructed in the ith dimension. The community indicators can be computed via the top eigenvectors of the utility matrix. This is equivalent to *optimizing the objective function over all the dimensions simultaneously*. As for modularity maximization, the average utility matrix in this case would be

$$\bar{M} = \frac{1}{d}\sum_{i=1}^{d} \widetilde{B}^{(i)} = \frac{1}{d}\sum_{i=1}^{d}\left\{\frac{A^{(i)}}{2m^{(i)}} - \frac{\mathbf{d}^{(i)}(\mathbf{d}^{(i)})^T}{(2m^{(i)})^2}\right\} \tag{21}$$

Finding out the top eigenvectors of the average utility matrix is equivalent to maximizing the average modularity as follows:

$$\max_S \frac{1}{d}\sum_{i=1}^{d} Tr(S^T \widetilde{B}^{(i)} S) = \max_S Tr(S^T \bar{M} S) \tag{22}$$

3.3 Feature integration

We can also perform the integration over the structural features extracted from each dimension of the network. One might conjecture that we can perform similar operations as we did for network interactions and utility matrices, i.e., taking the average of structural features as follows:

$$\bar{S} = \frac{1}{d}\sum_{i=1}^{d} S^{(i)} \tag{23}$$

Unfortunately, this straightforward extension does not apply to structural features. Because the solution S which optimizes the utility function is not unique. Dissimilar structural features do not suggest that the corresponding latent community structures are drastically different. For example, let S be the top-ℓ eigenvectors that maximize modularity Q, and V an orthonormal matrix such that

$$V \in \mathbb{R}^{\ell\times\ell}, \quad VV^T = I_\ell, \quad V^TV = I_\ell$$

It can be verified that SV also maximizes Q:

$$\frac{1}{2m}tr((SV)^T B(SV)) = \frac{1}{2m}tr(S^T BSVV^T) = \frac{1}{2m}tr(S^T BS) = Q_{\max}$$

Essentially, SV and S are equivalent under an orthogonal transformation. In the simplest case, $S' = -S$ is also a valid solution. Averaging these structural features does not result in sensible features.

Alternatively, we expect the structural features of different dimensions to be highly correlated *after certain transformations*. To capture the correlations between multiple sets of variables, (generalized) canonical correlation analysis (CCA) (Hotelling 1936; Kettenring 1971) is the standard statistical technique. CCA attempts to find a transformation for each set of variables such that the pairwise correlations are maximized. Here we briefly illustrate one scheme of generalized CCA which turns out to equal to principal component analysis (PCA) in our specific case.

Let $S^{(i)} \in \mathbb{R}^{n\times\ell}$ denote the structural features extracted from the ith dimension of the network, and $\mathbf{w_i} \in \mathbb{R}^{\ell}$ be the linear transformation applied to structural features of dimension i. The correlation between two sets of structural features after transformation is

$$R(i, j) = (S^{(i)}\mathbf{w_i})^T(S^{(j)}\mathbf{w_j}) = \mathbf{w_i}^T\left((S^{(i)})^T S^{(j)}\right)\mathbf{w_j} = \mathbf{w_i}^T C_{ij}\mathbf{w_j}$$

with $C_{ij} = (S^{(i)})^T S^{(j)}$ representing the covariance between the structural features of the ith and the jth dimensions. Generalized CCA attempts to maximize the summation of pairwise correlations as in the following form:

$$\max \sum_{i=1}^{d}\sum_{j=1}^{d}\mathbf{w_i}^T C_{ij}\mathbf{w_j} \tag{24}$$

$$s.t. \sum_{i=1}^{d}\mathbf{w_i}^T C_{ii}\mathbf{w_i} = 1 \tag{25}$$

Using standard Lagrange multiplier and setting the derivatives respect to $\mathbf{w_i}$ to zero, we obtain the equation below:

$$\begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1d} \\ C_{21} & C_{22} & \cdots & C_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ C_{d1} & C_{d2} & \cdots & C_{dd} \end{bmatrix}\begin{bmatrix}\mathbf{w_1}\\ \mathbf{w_2}\\ \vdots \\ \mathbf{w_d}\end{bmatrix} = \lambda \begin{bmatrix} C_{11} & 0 & \cdots & 0 \\ 0 & C_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & C_{dd} \end{bmatrix}\begin{bmatrix}\mathbf{w_1}\\ \mathbf{w_2}\\ \vdots \\ \mathbf{w_d}\end{bmatrix} \tag{26}$$

Recall that our structural features extracted from each dimension is essentially the top eigenvectors of the utility matrix satisfying$(S^{(i)})^T S^{(i)} = I$. Thus, matrix $diag(C_{11}, C_{22}, \ldots, C_{dd})$ in (26) becomes an identity matrix. Hence $\mathbf{w} = [\mathbf{w_1}^T, \mathbf{w_2}^T, \ldots, \mathbf{w_d}^T]^T$ corresponds to the top eigenvector of the full covariance matrix on the left side of (26), which is equivalent to PCA applied to data of the following form:

$$X = \left[S^{(1)}, S^{(2)}, \ldots, S^{(d)}\right] \tag{27}$$

```
Input: $Net = \{A^{(1)}, A^{(2)}, \cdots, A^{(d)}\}$,
       number of communities $k$,
       number of structural features to extract $\ell$;
Output: community assignment $idx$.
1. Compute top $\ell$ eigenvectors of the utility matrix as stated in Eq. (14);
2. Compute slim SVD of $X = [S^{(1)}, S^{(2)}, \cdots S^{(d)}] = UDV^T$;
3. Obtain lower-dimensional embedding $\widetilde{U} = U(:, k-1)$;
4. Normalize the rows of $\widetilde{U}$ to unit length;
5. Calculate the cluster $idx$ with k-means on $\widetilde{U}$.
```

Fig. 5 Algorithm: Structural feature integration for multi-dimensional networks

Suppose the SVD of X is $X = U\Sigma V^T$, then $\mathbf{w}$ corresponds to the first column of V. Thus we have

$$\frac{1}{d}\sum_{i=1}^{d} S^{(i)}\mathbf{w_i} = \frac{1}{d}\left[S^{(1)}, S^{(2)}, \ldots, S^{(d)}\right]\mathbf{w} = \frac{1}{d}XV_1 = \frac{\sigma_1}{d}U_1$$

Since σ_1/d is a scalar, U_1 is essentially the average feature values of each actor after we aggregate the structural features of different dimensions along with the transformation $\mathbf{w}$. There are $k-1$ degrees of freedom with k communities. To compute the $(k-1)$-dimension embedding, we just need to project the data X onto the top $(k-1)$ principal vectors. It follows that the top $(k-1)$ vectors of U are the aggregated structural features.

The detailed structural feature integration algorithm is summarized in Fig. 5. In summary, we first extract structural features from each dimension of the network via representative community detection methods; then PCA is applied on the concatenated data as in (27) to select the top eigenvectors. After projecting the data onto the principal vectors, we obtain a lower-dimensional embedding which captures the principal pattern across all the dimensions of the network. Then we can perform k-means on this embedding to find out the discrete community assignment.

3.4 Partition integration

Partition integration takes effect after the community partition of each network dimension is ready. This problem has been studied as the *cluster ensemble* problem (Strehl and Ghosh 2003; Asur et al. 2007), which combines multiple clustering results of the same data from a variety of sources into a single consensus clustering. Strehl and Ghosh (2003) propose three effective and comparable approaches: Cluster-Based Similarity Partitioning Algorithm (CPSA), HyperGraph Partition Algorithm and Meta-Clustering Algorithm. For brevity, we only present the basic idea of CPSA here. CPSA constructs a similarity matrix from each clustering. Two objects' similarity is 1 if they belong to the same group, 0 if they belong to different groups. Let $H^{(i)} \in \{0, 1\}^{n\times k}$ denote the community indicator matrix of clustering based on interactions at dimension i. The similarity between nodes can be computed as

$$\frac{1}{d}\sum_{i=1}^{d} H^{(i)}(H^{(i)})^T = \frac{1}{d}\sum_{i=1}^{d} \widetilde{H}\widetilde{H}^T \quad \text{where } \widetilde{H} = \left[H^{(1)}, H^{(2)}, \ldots, H^{(d)}\right]$$

Based on this similarity matrix between nodes, we can apply similarity-based community detection methods we introduced before to find out clusters. A disadvantage of this CPSA is that the computed similarity matrix can be dense, which might not be applicable to large networks. Instead, we can treat $\widetilde{H}$ as the feature representation of actors and cluster them based on k-means directly. Intuitively, if two actors are assigned to the same group in the majority of dimensions, they would share features. Thus, the two actors are likely to reside within the same community in the final consensus cluster as well.

3.5 Summary

In previous subsections, we have described different strategies to integrate multi-dimensional network information. Here, we summarize the pros and cons of different schemes.

Network integration, which simply averages the network interactions in different dimensions, can be problematic if the interactions are not comparable. In reality, actors often participate in different dimensions of a network with varied intensity. Even within the same group, the interaction can be very sparse in 1-D but relatively more observable in another dimension. So if there is 1-D with intensive interactions, simply averaging all the dimensions would overwhelm the structural information in other dimensions.

Utility integration sums up all the utility matrices. This combination is consistent with the overall objective. But it is unclear whether the utility function is directly comparable across different dimensions. For instance, in modularity maximization, the modularity is highly relevant to the density of interactions as well as the community structure. Is the average of utility function a reasonable choice? If not, how can we normalize it so that the utility in different dimensions are comparable. This will be studied more in the empirical study.

Feature integration identifies transformations such that the structural features of different dimensions become highly correlated. The transformations map structural features into the same space, thus aggregation is viable. Note that the extraction of structural features helps reduce the noise in each dimension of the network. Hence, feature integration is expected to be more robust compared with other methods.

Partition integration relies on discrete hard clusterings. Note that the classical k-means clustering algorithm normally finds a local optimal and is highly sensitive to the initial condition. Though k-means is applied to all the schemes, partition integration apply k-means to each dimension of the network to find out the partitions, which can introduce more uncertainty, hence are likely to yield results with relatively high variance.

4 Empirical study

We now discuss evaluation methods that are suitable for multi-dimensional networks. An ideal case is that we know *a priori* community memberships, or so-called ground truth. We can then adopt commonly used *normalized mutual information* (NMI) (Strehl and Ghosh 2003). Let π^a, π^b denote two different partitions of communities. NMI is defined as

$$NMI(\pi^a, \pi^b) = \frac{\sum_{h=1}^{k^{(a)}} \sum_{\ell=1}^{k^{(b)}} n_{h,\ell} \log\left(\frac{n \cdot n_{h,l}}{n_h^{(a)} \cdot n_\ell^{(b)}}\right)}{\sqrt{\left(\sum_{h=1}^{k^{(a)}} n_h^{(a)} \log \frac{n_h^a}{n}\right)\left(\sum_{\ell=1}^{k^{(b)}} n_\ell^{(b)} \log \frac{n_\ell^b}{n}\right)}}. \tag{28}$$

where n is the total number of data instances, $k^{(a)}$ and $k^{(b)}$ represent the numbers of communities in partitions π^a and π^b respectively, n_h^a, n_ℓ^b and $n_{h,\ell}$ are, respectively, the numbers of actors in the hth community of partition π^a, in the ℓth community of partition π^b, and in both the hth community of π^a and ℓth community of π^b. NMI is a measure between 0 and 1. NMI is equal to 1 when π^a, π^b are equivalent.

When ground truth is not available, an alternative evaluation method is needed to quantify extracted community structures employing different integration strategies. If a latent community structure is shared across network dimensions, we can perform CDNV as in Fig. 6. Given a multi-dimensional network $Net = \{A^{(i)} | 1 \leq i \leq d\}$, we can learn a community structure from $d - 1$ dimensions of the network and check how well the structure matches the left-out dimension ($A^{(p)}$). In other words, we use $d - 1$ dimensions for training and the remaining one for testing.

During the training, we obtain some communities (C), and use C to calculate the match of the community structure and the data of $A^{(p)}$. In order to measure the quality of the match, the utility function introduced in Sect. 2 can be used again, i.e., one can plug in block model approximation error, or normalized cut value as the quality function.

For consistency, we again use modularity as the quality measure:

$$Q = \frac{1}{2m} \sum_{C} \sum_{i \in C, j \in C} A_{ij} - \frac{d_i d_j}{2m}. \tag{29}$$

```
Input: $Net = \{A^{(1)}, A^{(2)}, \cdots, A^{(d)}\}$;
       a multi-dimensional integration scheme $f$;
Output: a community quality measure $Q^{(i)}$ for each network dimension.
1. for $p = 1, \ldots, d$
2.     hold out $A^{(p)}$ for testing;
3.     obtain community structure $H$ by applying $f$ to
        training dimensions $\{A^{(1)}, \cdots, A^{(p-1)}, A^{(p+1)}, \cdots, A^{(d)}\}$;
4.     compute the quality measure $Q^{(p)}$ based on $H$ and $A^{(p)}$.
5. end
```

Fig. 6 Algorithm: Cross-dimension network validation

A larger modularity implies more accurate community structure is discovered using the training data. The modularity value itself, however, must be treated with an ounce of caution. It has been shown that modularity has a resolution limit (Fortunato and Barthelemy 2007; Good et al. 2010). Modularity measure favors network partitions with groups or modules combined into a larger community. Thus, directly optimizing modularity typically leads to lesser number of communities than intuition. We also observed modularity values decreases with increasing number of communities in our experiments. In order to alleviate this resolution limit, one way (as done in Mucha et al. 2010) is to introduce a resolution parameter γ and modify the modularity function:

$$Q = \frac{1}{2m} \sum_{C} \sum_{i \in C, j \in C} A_{ij} - \gamma \frac{d_i d_j}{2m}. \tag{30}$$

Unfortunately, it is non-trivial to pick an appropriate resolution parameter. Alternatively, we fix the number of communities inside a network and then apply the standard modularity function in (29) to perform comparison. Since the resolution is kind of fixed in this case, we hope modularity can still be a good metric to compare the performance of different methods. Of course, one may plug in alterative quality function in the CDNV process.

We have to emphasize that the above two evaluation methods (NMI and CDNV) are designed for different contexts: NMI is suitable for data with known ground truth and CDNV for data without. If we could establish their relationship, we can then determine if CDNV can be used to compare different integration strategies for community detection. One way to establish the relationship between NMI and CDNV is to employ synthetic data.

To recap, we use Modularity Maximization to produce utility matrix M, compute structural features S via spectral analysis, and apply k-means to find community partitions[5] (Fig. 3). Different integration strategies can be applied at various stages as shown in Fig. 4. Baseline strategies are to not integrate $(d - 1)$ dimensions, but use single dimensions.

4.1 Experiments on synthetic data

The synthetic data has three groups, each having 50, 100, 200 members, respectively. There are four dimensions of interactions among these 350 members. For each dimension of the network, we sample within-group interaction probability for each group from a uniform distribution. Based on the within-group interaction probability, interactions occur between members following a Bernoulli distribution. Noise is also added by randomly connecting two nodes in the network. Since we have the group membership information for the synthetic data, NMI (28) can be employed to evaluate the performance of community detection.

[5] Since k-means clustering is sensitive to the initialization, we repeat k-means five times and pick whichever is the best as the community partition.

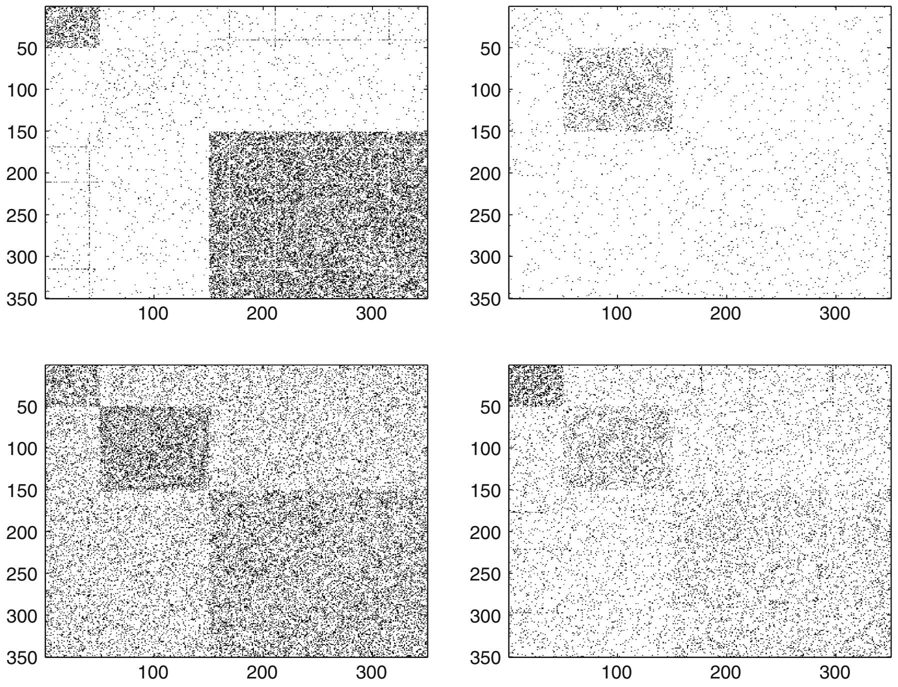

Fig. 7 One example of 4-dimensional network

Figure 7 shows one example of the generated multi-dimensional network. Clearly, different dimensions demonstrate different interaction patterns. Figure 8 reports the performance of community detection in terms of NMI, where $A1$, $A2$, $A3$ and $A4$ denote the performance based on a single dimension (or baseline strategies) and the other four bars show the performance of community detection using integration strategies corresponding to network (N), utility (U), feature (F) and partition (P) integration, respectively. Clearly, the four methods which integrate information from different network dimensions outperform those on single-dimensional networks. This could be easily explained by the patterns represented in Fig. 7. The first dimension of the network actually shows only two groups, and the second dimension involves only one group with the other two hidden behind the noise. Thus, using a single view is very unlikely to recover the correct latent community structure. This is indicated by the low NMI of the first two dimensions. Utilizing the information presented in all the dimensions, on the contrary, helps compensate each other and uncover the shared community structure.

Comparing different integration schemes, feature integration in this case, uncovers the true community information exactly, whereas the others do not. Figure 8 shows just one example. To conclude more confidently, we regenerate 100 different synthetic data sets and report the average performance of each method in Table 1. Clearly, multi-dimensional integration schemes outperform single-dimensional community detection methods in terms of NMI. Structural feature integration achieves the best performance

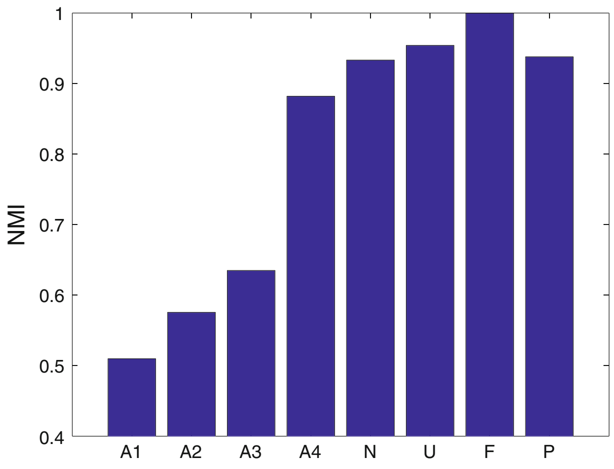

Fig. 8 Performance of different community detection methods. *A1*, *A2*, *A3* and *A4* show the performance on a single dimension. *N* Network, *U* utility, *F* feature, *P* partition integration

Table 1 Average performance over 100 runs

	Strategy	NMI	R_{NMI}	CDNV	R_{CDNV}
Single-dimensional	–	0.6903	5	0.1413	5
Multi-dimensional integration	Network	0.7946	4	0.1739	4
	Utility	0.9157	2	0.2035	2
	Feature	0.9351	1	0.2064	1
	Partition	0.8048	3	0.1785	3

NMI denotes the average performance of comparing the extracted communities with the latent group membership for generating a network, R_{NMI} the ranking comparing different strategies based on NMI, CDNV the performance based on CDNV, and R_{CDNV} the ranking based on CDNV. Note that NMI and CDNV yield consistent rankings

with lowest variance. This is because feature integration denoises the information presented in each dimension, thus is able to obtain a more robust clustering result. Network integration and utility integration, on the other hand, combine the noisy network or utility matrix directly, resulting in inferior performance. Partition integration relies on partitions extracted from each network dimension, and partitions depend on clustering algorithm being used. In our case, k-means clustering can produce local optimal partitions. Structural feature integration is the most effective approach among all of them.

In order to verify the validness of CDNV, we hold out one dimension for testing and pick the other three dimensions for training. The average performance of CDNV is also shown in the table. It is interesting that both evaluation schemes: NMI (with latent community membership information) and CDNV (without true community membership information) yield consistent rankings when comparing different strategies for community detection. Next, we will use CDNV to verify how different integration strategies work on real-world data.

4.2 Experiments on social media data

We now compare different multi-dimensional integration strategies using YouTube data. We discuss data collection and properties first and then report and analyze experimental findings.

4.2.1 YouTube data

YouTube[6] is currently the most popular video sharing web site. It is reported to "attract 100 million video views per day."[7] As of March 17th, 2008, there have been 78.3 million videos uploaded, with over 200,000 videos uploaded per day.[8] This social networking site allows users to interact with each other in various forms such as contacts, subscriptions, sharing favorite videos, etc. We use YouTube Data API[9] to crawl the contacts network, subscription network as well as each user's favorite videos. We choose 100 authors who recently uploaded videos as the seed set for crawling, and expand the network via their contacts and subscriptions. We obtain a small portion of the whole network, with 30,522 user profiles reaching in total 848,003 contacts and 1,299,642 favorite videos. After removing those users who decline to share their contact information, we have 15,088 active user profiles as presented in three different interactions: two adjacency matrices of size $15{,}088 \times 848{,}003$ representing contact relationship and subscriptions, and a matrix of size $15{,}088 \times 1{,}299{,}642$ representing users' favorite videos.

One issue is that the collected subscription network is directional while most community detection methods such as block models, spectral clustering and modularity maximization, are proposed for undirected networks. For such cases, simply ignoring the direction can confuse the two roles of the directional interaction. Instead, we decompose the asymmetric interaction A into two unidirectional interactions:

$$A' = AA^T; \tag{31}$$

$$A'' = A^T A. \tag{32}$$

Essentially, if two social actors both subscribe to the same set of users, it is likely that they are similar and share the same community; On the other hand, if two are referred by the same set of actors, their similarity tends to be higher than that of random pairs. This is similar to the two roles of hub and authority of web pages as mentioned in Kleinberg (1999).

To utilize all aspects of information in our collected data, we construct a 5-dimensional network:

$A^{(1)}$ contact network: the contact network among those 15,088 active users;

[6] http://www.youtube.com/.

[7] http://www.usatoday.com/tech/news/2006-07-16-youtube-views_x.htm.

[8] http://ksudigg.wetpaint.com/page/YouTube+Statistics?t=anon.

[9] http://code.google.com/apis/youtube/overview.html.

Table 2 The density of each dimension in the constructed 5-dimensional network

Network	Dimension	Density
$A^{(1)}$	Contact	6.74×10^{-4}
$A^{(2)}$	Co-contact	1.71×10^{-2}
$A^{(3)}$	Co-subscription	4.90×10^{-2}
$A^{(4)}$	Co-subscribed	1.97×10^{-2}
$A^{(5)}$	Favorite	3.34×10^{-2}

$A^{(2)}$ co-contact network: two active users are connected if they both add another user as contact; This is constructed based on all the reachable 848,003 users (excluding those active ones) in our collected data following Eq. 31.

$A^{(3)}$ co-subscription network: the connection between two users denotes they subscribe to the same user; constructed following Eq. 31;

$A^{(4)}$ co-subscribed network: two users are connected if they are both subscribed by the same user; constructed following Eq. 32;

$A^{(5)}$ favorite network: two users are connected if they share favorite videos.

All these different interactions are correlated with user interests. According to homophily effect well studied in social science (McPherson et al. 2001), people tend to connect to others sharing certain similarities. Thus, we expect that connected friends in the contact network $A^{(1)}$ is more likely to share certain interests. Similarly, if both users connect to another user or a favorite video (as $A^{(2)}$, $A^{(3)}$ or $A^{(5)}$), they are likely to share certain interests. On the other hand, if two users are subscribed by the same set of users (as in $A^{(4)}$), their shared content, thus their interests, are similar. Essentially, we hope to extract communities sharing similar interests by integrating heterogeneous interactions.

Table 2 shows the connection density of each dimension. Contact dimension is the most sparse one, while the other dimensions, due to the construction, are denser. Figure 9 shows the degree distribution in contacts network and favorite network. Both follow a power law pattern (Clauset et al. 2007) as expected.

4.2.2 *Comparative study*

The four multi-dimensional integration schemes as well as community detection methods on a single dimension are compared. We cluster actors involved in the network into different numbers of communities. The clustering performance of single-dimensional and multi-dimensional methods when $k = 20$, 40 and 60 are presented in Table 3. In the table, rows represent methods and columns denote the rankings when a certain network dimension is used as test data. The bold face denotes the best performance in each column for each case. Note that in our CDNV, the test dimension is not available during training, thus the diagonal entries for single-dimensional methods are not shown.

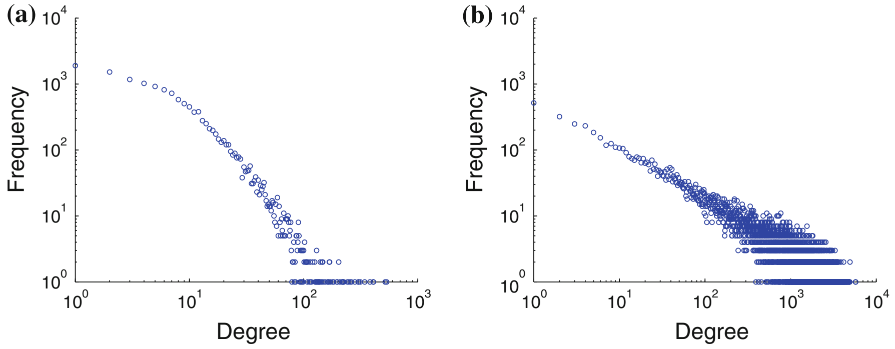

Fig. 9 Power law distribution on different dimensions. **a** Contact dimension and **b** favorite dimension

Feature integration is clearly the winner most of the time, except for certain rare cases (e.g., using $A^{(4)}$ as the test dimension when $k = 40$ or 60). We notice that the rankings of different integration strategies do not change much with different k. A closer examination reveals that utilizing information of all the dimensions (except network integration) outperforms single-dimensional clustering. Network integration does not work well, because the network studied here are weighted and simple average blurs the latent community structure information presented in each dimension. In terms of performance ranking, feature integration $\prec$ utility integration $\prec$ partition integration $\prec$ network integration. Feature integration, by removing noise in each dimension, yields the most accurate community structure among all the methods.

5 Further analysis

In the previous section, we have demonstrated that feature integration tends to outperform other integration schemes. In this section, we perform further analysis concerning the computational time of different methods, normalization effect during integration, sensitivity to related parameters, and alternative community detection methods for integration.

5.1 Efficiency study

The four multi-dimensional integration schemes differ drastically over time complexity. Table 4 summarizes the asymptomatic time complexity of different methods. Clearly, network integration and utility integration are the most efficient, which require the average of network matrix or utility matrix with time complexity $O(dm)$. Following that, one instance of eigenvector computation and k-means clustering are required. Feature integration and partition integration require the computation of structural features in each network dimension. Note that this can be accelerated via parallel computing. Feature integration needs to compute the SVD of a dense matrix X (27) of size $n \times d\ell$, which costs $O(nd\ell \cdot \min(n, d\ell))$. Since $d \ll n$ and $\ell \ll n$. The additional computational cost is still acceptable. Partition integration, without SVD,

Table 3 Performance when actors are clustered into 20, 40, and 60 communities, respectively

	Strategies	$R^{(1)}$	$R^{(2)}$	$R^{(3)}$	$R^{(4)}$	$R^{(5)}$	$R_{average}$
$k = 20$							
Single-dimensional community detection	$A^{(1)}$	–	7	8	8	8	7.75
	$A^{(2)}$	4	–	5	5	6	5.00
	$A^{(3)}$	6	5	–	4	4	4.75
	$A^{(4)}$	7	4	4	–	5	5.00
	$A^{(5)}$	8	6	6	6	–	6.50
Multi-dimensional integration	Network	5	8	7	7	7	6.80
	Utility	2	2	2	2	2	2.00
	Feature	**1**	**1**	**1**	**1**	**1**	**1.00**
	Partition	3	3	3	3	3	3.00
$k = 40$							
Single-dimensional community detection	$A^{(1)}$	–	8	6	7	8	7.75
	$A^{(2)}$	4	–	4	5	6	4.75
	$A^{(3)}$	5	4	–	4	4	4.25
	$A^{(4)}$	7	6	5	–	7	6.25
	$A^{(5)}$	8	7	7	6	–	7.00
Multi-dimensional integration	Network	6	5	8	8	5	6.40
	Utility	2	2	2	3	2	2.20
	Feature	**1**	**1**	**1**	2	**1**	**1.20**
	Partition	3	3	3	**1**	3	2.60
$k = 60$							
Single-dimensional community detection	$A^{(1)}$	–	5	6	7	8	6.50
	$A^{(2)}$	3	–	5	4	6	4.50
	$A^{(3)}$	6	6	–	5	7	6.00
	$A^{(4)}$	7	4	4	–	5	5.00
	$A^{(5)}$	8	8	7	6	–	7.25
Multi-dimensional integration	Network	5	7	8	8	4	6.40
	Utility	2	2	2	**1**	2	1.80
	Feature	**1**	**1**	**1**	2	**1**	**1.20**
	Partition	4	3	3	3	3	3.20

In the table, $R^{(i)}(1 \leq i \leq 5)$ denotes the ranking of each method based on CDNV as using $A^{(i)}$ for testing, and $R_{average}$ the average ranking across all network dimensions. *Bold* entries denote the best in each column for each case

requires many more runs of k-means clustering, but without the SVD computation for integration. Since the major computational cost is associated with the eigenvector problem, feature integration and partition integration are expected to take more time.

Figures 10 and 11 show the computational time of modularity maximization with respect to a variety of community numbers and network sizes. In both figures, feature

Table 4 Time complexity of different integration strategies

Integration scheme	Integration cost	$\#T_{utility}$	$\#T_{eig}$	$\#T_{kmeans}$
Network integration	$O(dm)$	1	1	1
Utility integration	$O(dm)$	d	1	1
Feature integration	$O(nd\ell \cdot \min(n, d\ell))$	d	d	1
Partition integration	T_{kmeans}	d	d	d

Integration cost denotes the additional cost to perform the integration. $\#T_{utility}$, $\#T_{eig}$, and $\#T_{kmeans}$ denote the required number of utility matrix construction, eigenvector computation, and kmeans clustering, respectively. $T_{utility}$, T_{eig} and T_{kmeans} are specified in (15), (16) and (17)

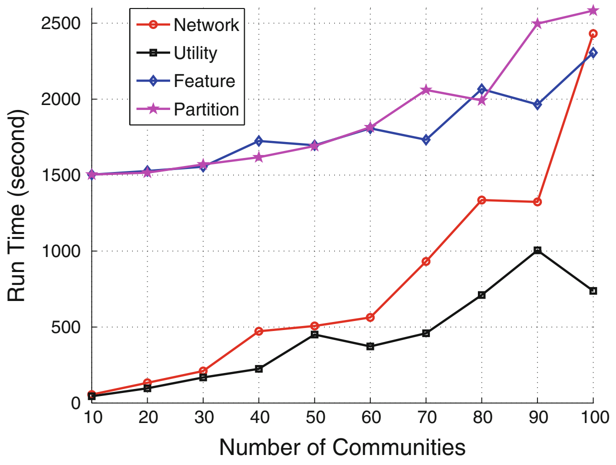

Fig. 10 Computation time with respect to varying number of communities on YouTube of 15,088 nodes

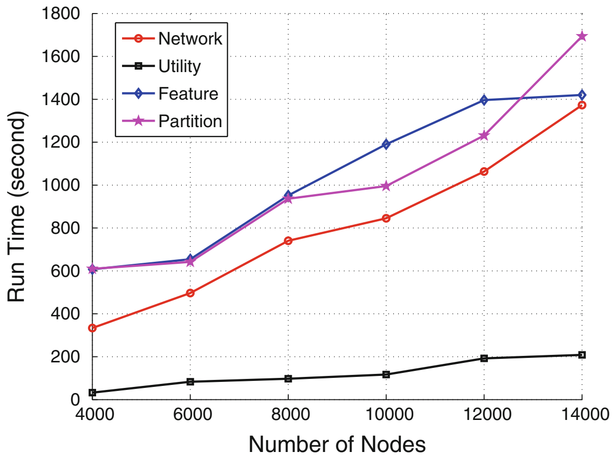

Fig. 11 Computation time with respect to varying network size (the number of communities is fixed to 40)

integration and partition integration are comparable, which is consistent with our analysis. By contrast, network integration and utility integration need to compute the eigenvector of only one utility matrix, thus it is more efficient. However, as we have demonstrated in the previous section, the performance of these two strategies is not comparable to feature integration. Note that in Table 4, we only show the asymptotic computation time. In reality, the network density can also affect the computational cost. It is observed when the number of clusters is huge, the computation time of integrated network even takes more time than feature integration (100 communities in Fig. 10). As shown in Fig. 11, the computational time scales linearly with respect to network size, promising for applications to large-scale networks. In summary, if efficiency is the major concern, we recommend utility integration, which is fastest, and is only second in performance to the optimal integration scheme. Otherwise, feature integration, though with additional computational cost, should be selected.

5.2 Normalization effect

In the previous section, we showed that network integration and utility integration is not comparable to feature integration. One conjecture is that whether or not we can find an effective normalization or weighting scheme such that they can be integrated more effectively. Here, as an attempt, we try some straightforward schemes and show the effect.

For network integration, a natural solution is to normalize the interaction by the total number of interactions. Specifically, we have the following weighted network integration:

$$\bar{A} = \sum_{i=1}^{d} A^{(i)}_{normalized} = \sum_{i=1}^{d} A^{(i)}/(2m^{(i)}) \tag{33}$$

Essentially, after this normalization, the total weight of the interaction becomes 1 in each dimension.

As for utility integration, one hypothesis is to use the community strength in each dimension as a guide to do the weighted average. If one dimension's community structure is more prominent, it seems reasonable to trust that dimension more. Let $Q^{(i)}$ denote the modularity computed in dimension i. We integrate the utility matrix in a weighted fashion as follows:

$$\bar{M} = \sum_{i=1}^{d} Q^{(i)} M^{(i)} \tag{34}$$

Due to the space limit, we only show the performance on the contact dimension of the YouTube network in Fig. 12. The attempt of normalizing network interactions helps most of the time, and utility weighting shows comparable performance to simple average of utility. It seems assigning different weights to the dimensions requires

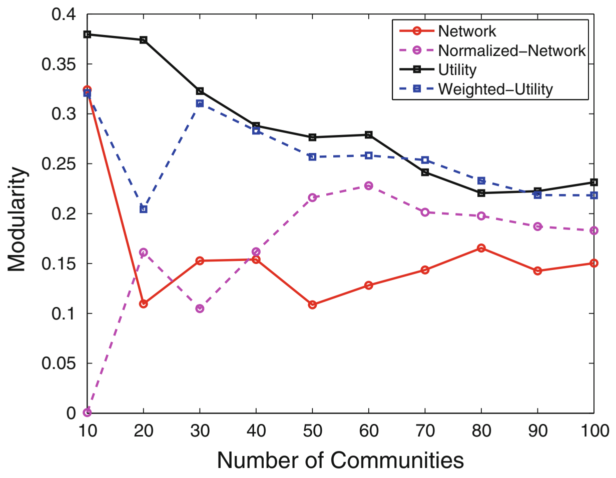

Fig. 12 Performance with normalization (weighting) schemes

more insightful understanding upon the dimensions. After all, the performance of network integration and utility integration after normalization and weighting is still not comparable to feature integration.

5.3 Sensitivity of feature integration

In feature integration, one parameter is the number of structural features to extract (ℓ in Fig. 5). In this part, we study the performance sensitivity of feature integration with respect to this parameter. We vary the number of structural features from 10 to 500 and show the performance variations in Fig. 13. The performance stabilizes when reasonable large number of structural features (say, >150) are extracted. As long as there are enough structural features, the performance is reasonably good. That is, feature integration is not sensitive to the parameter in a large range. In practice, we can start from a reasonably large number, and exploit CDNV to select a proper parameter.

One minor note is that the modularity value, as demonstrated in the figure, is negatively correlated with the number of communities, which echoes our discussion of the resolution limit of modularity at the beginning of Sect. 4.

5.4 Alternative community detection methods

Note that this work presents a general framework to integrate information of heterogeneous interactions. Previously, we showed the result based on modularity maximization. The same integration schemes can be applied to other community detection methods as well, such as block model approximation and spectral clustering. The latent space model is not included here due to its high computational cost as discussed in Sect. 2.5. We can simply replace the utility matrix with network interaction or graph

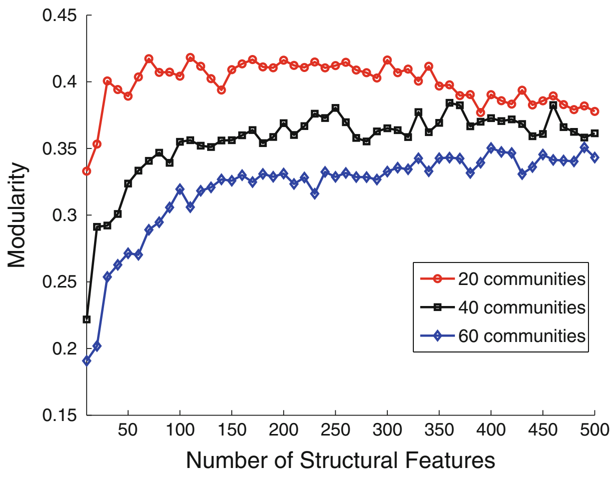

Fig. 13 Sensitivity to number of structural features

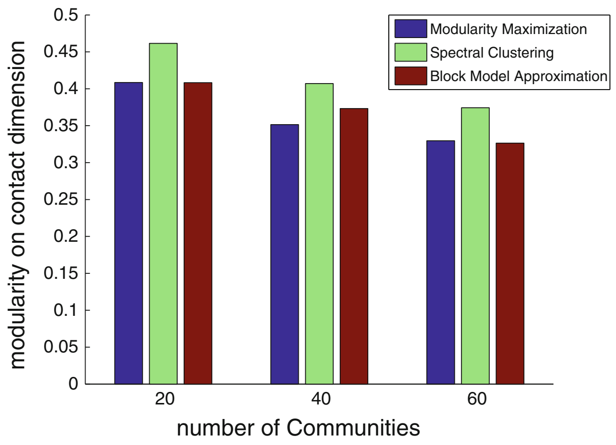

Fig. 14 Performance of different community detection methods

Laplacian as specified in (14). One interesting question is which community detection method is the best?

Here, we combine the feature integration strategy with block model approximation, spectral clustering and modularity maximization, respectively. The resultant performance on the contact dimension of YouTube network is plotted in Fig. 14. Spectral clustering is consistently better than modularity maximization and block model approximation. This result is consistent with that as reported in White and Smyth (2005).

Figure 15 shows the performance of four different integration schemes with spectral clustering. Clearly, feature integration, similar to the case of modularity maximization, is the winner. Note that our integration scheme is independent of the community

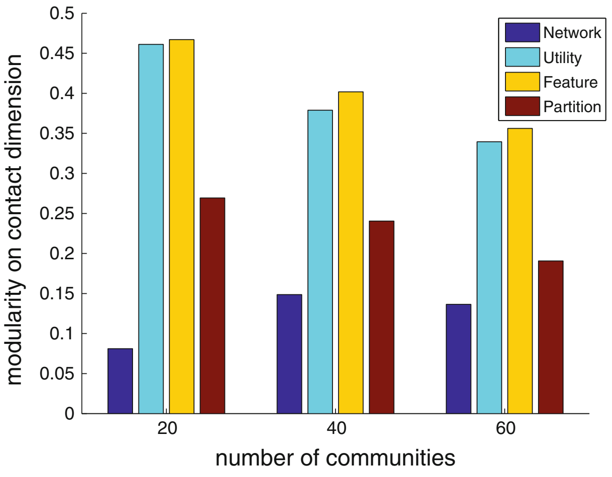

Fig. 15 Performance of integration schemes with spectral clustering

detection method. With a proper constructed utility matrix, we might achieve better performance.

6 Related work

6.1 Applications

Multi-dimensional networks (or multiple networks constructed from disparate sources) appear in many web applications. Aleman-Meza et al. (2006) constructs a semantic web to detect the Conflict of Interest relationship among paper reviewers and authors. Two social networks FOAF (Friend-of-a-Friend) and DBLP (co-author) networks are integrated in terms of attribute similarity of persons. Jung and Euzenat (2007) construct a semantic social network which includes a social network, an ontology network and a concept network. It is shown that relationships in one network might be inferred from another. Yingzi Jin and Ishizuka (2008) study the entity ranking problem in social networks. The authors first extract different relations between entities and construct heterogeneous social networks. Then they integrate the constructed networks with different weighting methods for more accurate ranking. Zhou et al. (2008) recommend documents in a digital library by integrating a citation network and networks of documents and other related entities.

6.2 Cluster ensemble, consensus clustering and multiview clustering

Some work attempts to address unsupervised learning with multiple data sources or clustering results, such as *cluster ensemble* (Strehl and Ghosh 2003; Topchy et al. 2003; Fern and Brodley 2004) and *consensus clustering* (Monti et al. 2003; Hu and Sung 2005; Nguyen and Caruana 2007; Goder and Filkov 2008). These methods essentially fall into partition integration scheme presented in our framework. Most of the algorithms aim to find a robust clustering based on multiple clustering results, which are prepared via feature or instance sampling or disparate clustering algorithms. A similar idea is applied to community detection in social networks (Hopcroft et al. 2003). A small portion of connections between nodes are randomly removed before each run, leading to multiple different clustering results. Those clusters occurring repeatedly are considered more stable, and are deemed to reflect the natural communities in reality. However, all the cluster ensemble methods concentrate on either attribute-based data or one-dimensional networks.

Another related field is multi-view clustering. Bickel and Scheffere (2004) propose co-EM and an extension of k-means and hierarchical clustering to handle data with two conditional independent views. de Sa (2005) creates a bipartite based on the two views and tries to minimize the disagreement. Different spectral frameworks with multiple views are studied in Zhou and Burges (2007) and Long et al. (2008). The former defines a weighted mixture of random walk over each view to identify communities. The latter assumes clustering membership of each view is provided and finds an optimal community pattern via minimizing the divergence of the transformed optimal pattern and the community membership of each view. A variant of utility integration based on block model approximation plus regularization is presented in Tang et al. (2009b). Similarly, Argyriou et al. (2006) suggests combining graph Laplacians for semi-supervised learning. It is empirically verified that our proposed integration schemes also apply to spectral clustering and block model approximation, and feature integration tends to be the most robust one.

Some theoretical analysis of multi-view clustering via CCA is presented in Chaudhuri et al. (2009). It shows that under the assumption that the views are uncorrelated given the cluster label, a much weaker condition is required for CCA to separate clusters successfully. However, the conclusion is based on two views with each being attributes. It requires further research to generalize the theoretical result to networks of multiple heterogeneous interactions.

6.3 Integration of networks of multiple relations

It is noticed that some generative models have been proposed to find the shared latent community structure with multiple type of relations among entities. Infinite relational model (Kemp et al. 2006) is similar to the block model. It differs by imposing a Chinese restaurant process over the latent variable for community assignment which allows the model to automatically determine the number of communities. Latent feature model (Miller et al. 2009) echoes our work by introducing latent features (similar to the idea of our structural features) to explain interactions between entities, and imposing an Indian buffet process as a prior to automatically determine the number of latent features. All these models can be very naturally extended to networks of multiple relations. All methods basically assume multiple relations to share the same latent feature or soft cluster assignment. The community structure can be determined by maximizing the likelihood. In essence, it is quite similar to the approach we mentioned as utility integration, and the utility there is the likelihood defined by the models.

Asur et al. (2007) address the clustering problem in protein–protein interaction networks. The authors define different metrics (clustering coefficient and edge betweenness) to define node similarity, and apply different clustering methods first to obtain disparate community partition first. Based on that, a PCA is applied to identify the shared and significant communities. This method inherently the partition integration. But it adds PCA as a preprocessing step to reduce the correlation and redundancy of community partitions. As we discussed in the empirical analysis, partition integration does not guarantee a better performance than other type of integration, yet its time complexity can be very expensive.

Recent research demonstrates intense interest towards more complicated multi-dimensional networks. One such example is time-dependent multiplex networks (Lin et al. 2008; Mucha et al. 2010). That is, the interaction becomes time-independent. In such a case, additional knowledge can be utilized to find the community structure at each timestamp. It is assumed that the community of neighboring timestamps should be similar. Thus, the problem boils down to finding a smoothly evolving community sequence over time. Another line of seemingly relevant research is to find communities with multiple type of entities (Tang et al. 2008; Lin et al. 2009). These works differs from our work. For certain applications and problems, it is desirable to find a shared community structure, such as the motivating example described in the introduction. We believe the framework and the two-stage approach proposed in this work pave the way to help address other related problems as well.

7 Conclusions and future work

Multi-dimensional networks commonly exist in many social networking sites, reflecting diverse individual activities. In this work, we propose and discuss different strategies to detect the latent communal structure in a multi-dimensional network. We formally describe the community detection problem in multi-dimensional networks and present a framework of different integration schemes to handle the problem. We show that representative community detection methods such as latent space models, block model approximation, spectral clustering, and modularity maximization, can be presented in a unified view involving four components: network interactions, utility matrix, structural features and community partitions. In this way, we can integrate the information presented in different network dimensions in terms of each component, leading to four different integration schemes: network integration, utility integration, feature integration and partition integration. We systematically study these different integration schemes and show that feature integration, which extracts structural features from each dimension of a multi-dimensional network and integrate them via PCA, outperforms other integration schemes.

As we have shown in the empirical study, utility integration is efficient when compared with feature integration. However, its performance depends on a clever weighting scheme over each dimension. It is intriguing to find an effective scheme that can boost the performance of utility integration as comparable to feature integration while maintaining efficiency. In our current work, we assume that heterogeneous interactions share the same community structure. When community structures vary significantly in subsets of dimensions, new research questions arise. Can we automatically determine which dimensions share the same community structure? How are they correlated? By CDNV, we might be able to calibrate the correlation between different network dimensions. However, this problem becomes complicated if some communities are shared across different dimensions whereas others are not. Further research is required in this area. It would also be interesting to extend the integration strategies to handle overlapping communities to construct a semantic ontology from tag networks. We expect that more research on community detection in multi-dimensional networks will emerge in the near future.

Acknowledgments This work was, in part, supported by Air Force Office of Scientific Research (AFOSR) and Office of Naval Research (ONR) grants.

References

Abou-Rjeili A, Karypis G (2006) Multilevel algorithms for partitioning power-law graphs. In: Proceedings of the 20th international conference on parallel and distributed processing (IPDPS'06). IEEE Computer Society, Washington, DC, p 124

Airodi EM, Blei D, Fienberg SE, Xing EP (2008) Mixed membership stochastic blockmodels. J Mach Learn Res 9:1981–2014

Aleman-Meza B, Nagarajan M, Ramakrishnan C, Ding L, Kolari P, Sheth AP, Arpinar BI, Joshi A, Finin T (2006) Semantic analytics on social networks: experiences in addressing the problem of conflict of interest detection. In: WWW '06: proceedings of the 15th international conference on World Wide Web. ACM Press, New York, pp 407–416

Argyriou A, Herbster M, Pontil M (2006) Combining graph Laplacians for semi-supervised learning. Adv Neural Inf Process Syst 18:67

Asur S, Parthasarathy S, Ucar D (2007) An event-based framework for characterizing the evolutionary behavior of interaction graphs. In: KDD '07: proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 913–921

Backstrom L, Huttenlocher D, Kleinberg J, Lan X (2006) Group formation in large social networks: membership, growth, and evolution. In: KDD '06: proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 44–54

Bansal N, Chiang F, Koudas N, Tompa FW (2007) Seeking stable clusters in the blogosphere. In: VLDB '07: proceedings of the 33rd international conference on very large data bases. Vienna, Austria, pp 806–817

Bickel S, Scheffere T (2004) Multi-view clustering. In: ICDM '04: proceedings of the fourth IEEE international conference on data mining. IEEE Computer Society, Washington, DC, pp 19–26

Borg I, Groenen P (2005) Modern multidimensional scaling: theory and applications. Springer, New York

Chaudhuri K, Kakade SM, Livescu K, Sridharan K (2009) Multi-view clustering via canonical correlation analysis. In: ICML '09: proceedings of the 26th annual international conference on machine learning. ACM, New York, pp 1–8

Clauset A, Newman M, Moore C (2004) Finding community structure in very large networks. Arxiv preprint cond-mat/0408187

Clauset A, Shalizi CR, Newman MEJ (2007) Power-law distributions in empirical data. arXiv, 706

de Sa VR (2005) Spectral clustering with two views. In: Proceedings of workshop of learning with multiple views. Bonn, Germany, pp 20–27

Demmel J, Dongarra J, Ruhe A, van der Vorst H (2000) Templates for the solution of algebraic eigenvalue problems: a practical guide. Society for Industrial and Applied Mathematics, Philadelphia

Fern XZ, Brodley CE (2004) Solving cluster ensemble problems by bipartite graph partitioning. In: ICML '04: proceedings of the twenty-first international conference on machine learning. ACM, New York, p 36

Fortunato S (2010) Community detection in graphs. Phys Rep 486(3–5):75–174
Fortunato S, Barthelemy M (2007) Resolution limit in community detection. Proc Natl Acad Sci USA 104(1):36–41
Goder A, Filkov V (2008) Consensus clustering algorithms: comparison and refinement. In: Proceedings of the workshop on algorithm engineering and experiments, ALENEX 2008, San Francisco, CA, January 19, 2008
Good B, De Montjoye Y, Clauset A (2010) Performance of modularity maximization in practical contexts. Phys Rev E 81(4):046106
Guimera R, Amaral LAN (2005) Functional cartography of complex metabolic networks. Nature 433:895–900
Hopcroft J, Khan O, Kulis B, Selman B (2003) Natural communities in large linked networks. In: KDD '03: proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 541–546
Hotelling H (1936) Relations between two sets of variates. Biometrika 28(3/4):321–377
Hu T, Sung SY (2005) Consensus clustering. Intell Data Anal 9(6):551–565
Jung J, Euzenat J (2007) Towards semantic social networks. In: Proceedings of the European semantic Web conference (ESWC2007), volume 4519 of LNCS. Springer, Berlin, pp 267–280
Kang H, Getoor L, Singh L (2007) Visual analysis of dynamic group membership in temporal social networks. SIGKDD Explor Spec Issue Vis Anal 9(2):13–21
Kemp C, Tenenbaum J, Griffiths T, Yamada T, Ueda N (1999/2006) Learning systems of concepts with an infinite relational model. In: Proceedings of the national conference on artificial intelligence, vol 21. AAAI Press/MIT Press, Menlo Park/Cambridge, p 381
Kettenring J (1971) Canonical analysis of several sets of variables. Biometrika 58:433–451
Kleinberg JM (1999) Authoritative sources in a hyperlinked environment. J ACM 46(5):604–632
Lancichinetti A, Fortunato S (2009) Community detection algorithms: a comparative analysis. Phys Rev E 80(5):056117
Leskovec J, Lang KJ, Mahoney M (2010) Empirical comparison of algorithms for network community detection. In: WWW '10: proceedings of the 19th international conference on World Wide Web. ACM, New York, pp 631–640
Lin Y-R, Chi Y, Zhu S, Sundaram H, Tseng BL (2008) Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. In: Proceeding of the 17th international conference on World Wide Web (WWW '08). ACM, New York, NY, pp 685–694
Lin Y-R, Chi Y, Zhu S, Sundaram H, Tseng BL (2009) Analyzing communities and their evolutions in dynamic social networks. ACM Trans Knowl Discov Data 3(2):1–31
Lizorkin D, Medelyan O, Grineva M (2009) Analysis of community structure in Wikipedia. In: WWW '09: proceedings of the 18th international conference on World Wide Web. ACM, New York, pp 1221–1222
Long B, Yu PS, Zhang ZM (2008) A general model for multiple view unsupervised learning. In: Proceedings of the 2008 SIAM international conference on data mining, pp 822–833
Luxburg Uv (2007) A tutorial on spectral clustering. Stat Comput 17(4):395–416
McPherson M, Smith-Lovin L, Cook JM (2001) Birds of a feather: homophily in social networks. Annu Rev Sociol 27:415–444
Mika P (2007) Ontologies are us: a unified model of social networks and semantics. In: Web semantics: science, services and agents on the World Wide Web, vol 5, no 1, pp 5–15. Selected papers from the international semantic Web conference (ISWC2005)
Miller K, Griffiths T, Jordan M (2009) Nonparametric latent feature models for link prediction. Adv Neural Inf Process Syst 22:1276–1284
Monti S, Tamayo P, Mesirov J, Golub T (2003) Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. Mach Learn 52(1–2):91–118
Mucha P, Richardson T, Macon K, Porter M, Onnela J (2010) Community structure in time-dependent, multiscale, and multiplex networks. Science 328(5980):876–878
Newman M (2006a) Finding community structure in networks using the eigenvectors of matrices. Phys Rev E 74(3):036104
Newman M (2006b) Modularity and community structure in networks. Proc Natl Acad Sci USA 103(23):8577–8582
Nguyen N, Caruana R (2007) Consensus clusterings. In: ICDM '07: proceedings of the 2007 seventh IEEE international conference on data mining. IEEE Computer Society, Washington, DC, pp 607–612
Palla G, Barabasi A-L, Vicsek T (2007) Quantifying social group evolution. Nature 446(7136):664–667

Richardson M, Domingos P (2002) Mining knowledge-sharing sites for viral marketing. In: KDD '02: proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, New York, pp 61–70

Rosvall M, Bergstrom C (2008) Maps of random walks on complex networks reveal community structure. Proc Natl Acad Sci USA 105(4):1118–1123

Sarkar P, Moore AW (2005) Dynamic social network analysis using latent space models. SIGKDD Explor Newsl 7(2):31–40

Specia L, Motta E (2007) Integrating folksonomies with the semantic web. In: Proceedings of the 4th European conference on the semantic Web: research and applications. Springer, Berlin, pp 624–639

Strehl A, Ghosh J (2003) Cluster ensembles—a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3:583–617

Tang L, Liu H (2009a) Relational learning via latent social dimensions. In: KDD '09: proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 817–826

Tang L, Liu H (2009b) Scalable learning of collective behavior based on sparse social dimensions. In: CIKM '09: proceeding of the 18th ACM conference on information and knowledge management. ACM, New York, pp 1107–1116

Tang L, Liu H, Zhang J, Nazeri Z (2008) Community evolution in dynamic multi-mode networks. In: KDD '08: proceeding of the 14th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 677–685

Tang L, Wang X, Liu H (2009a) Uncovering groups via heterogeneous interaction analysis. In: Proceeding of IEEE international conference on data mining. IEEE Computer Society, Washington, DC, pp 503–512

Tang W, Lu Z, Dhillon IS (2009b) Clustering with multiple graphs. In: ICDM '09: proceedings of the 2009 ninth IEEE international conference on data mining. IEEE Computer Society, Washington, DC, pp 1016–1021

Topchy A, Jain AK, Punch W (2003) Combining multiple weak clusterings. In: ICDM '03: proceedings of the third IEEE international conference on data mining. IEEE Computer Society, Washington, DC, p 331

Wasserman S, Faust K (1994) Social network analysis: methods and applications. Cambridge University Press, Cambridge

White S, Smyth P (2005) A spectral clustering approach to finding communities in graphs. In: Proceedings of the fifth SIAM international conference on data mining, pp 274–285

Witten IH, Frank E (2005) Data mining: practical machine learning tools and techniques. Morgan Kaufman, San Francisco

Yingzi Jin YM, Ishizuka M (2008) Ranking entities on the web using social network mining and ranking learning. In: WWW '08: proceedings of the 17th international conference on World Wide Web. ACM Press, New York, pp 21–25

Yu K, Yu S, Tresp V (2005) Soft clustering on graphs. In: Advances in neural information processing systems 18. MIT Press, Cambridge, pp 1553–1560

Zhou D, Burges CJC (2007) Spectral clustering and transductive learning with multiple views. In: ICML '07: proceedings of the 24th international conference on machine learning. ACM, New York, 1159–1166

Zhou D, Zhu S, Yu K, Song X, Tseng BL, Zha H, Giles CL (2008) Learning multiple graphs for document recommendations. In: WWW '08: proceeding of the 17th international conference on World Wide Web. ACM, New York, pp 141–150