

## Community Detection in Multi-Dimensional Networks

Lei Tang · Xufei Wang · Huan Liu

Received: date / Accepted: date

**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 procedure to compare the performance of different methods. 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.

---

Lei Tang, Xufei Wang, Huan Liu  
Computer Science and Engineering, Arizona State University, Tempe, AZ 85287, USA  
E-mail: {L.Tang, Xufei.Wang, Huan.Liu@asu.edu}

## 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 [41]. The extracted communities can be utilized for further analysis such as visualization [17], viral marketing [29], determining the causal factors of group formation [3], detecting group evolution [28] or stable clusters [4], relational learning [33], and building ontology for semantic web [23,31,20].

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*<sup>1</sup> 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 Figure 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 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

---

<sup>1</sup> 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 network [35]. Here, we use multi-dimensional network to emphasize that actors are involved in disparate interactions.

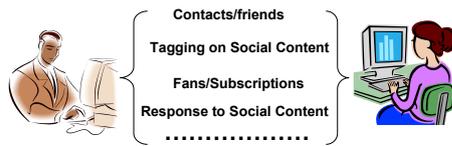


Fig. 1: Multi-Dimensional Network

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.

Integrating assorted forms of interaction can compensate for incomplete information in each dimension as well as reduce the 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 cross-dimension network validation 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 one-dimensional 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

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

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 [43]. One representative approach is *multi-dimensional scaling* (MDS) [6]. 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* [41], 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  $\ell$ -dimensional space such that  $S$  are column orthogonal. It can be shown [6, 30] that

$$SS^T \approx -\frac{1}{2}(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)(P \circ P)(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T) = \tilde{P} \quad (2)$$

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  $\tilde{P}$  and  $SS^T$  as follows:

$$\min \|SS^T - \tilde{P}\|_F^2 \quad (3)$$

Suppose  $V$  are the top  $\ell$  eigenvectors of  $\tilde{P}$  with largest eigenvalues,  $\Lambda$  a diagonal matrix of top  $\ell$  eigenvalues  $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_\ell)$ . The optimal  $S$  is  $S =$

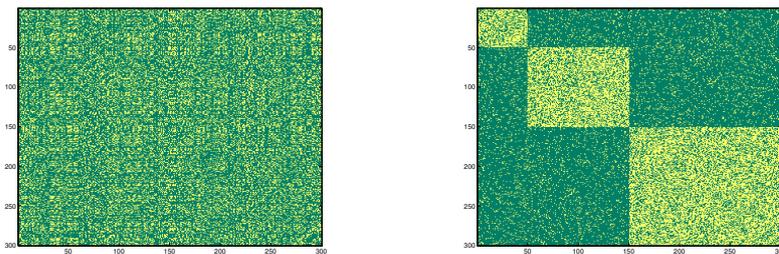


Fig. 2: Basic Idea of Block Model Approximation

$V\Lambda^{\frac{1}{2}}$ . Note that this multi-dimensional scaling corresponds to an eigenvector problem of matrix  $\tilde{P}$ . 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 Figure 2 where the left graph shows a network and 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:

$$A \approx S\Sigma S^T \quad (4)$$

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

$$\min \|A - S\Sigma S^T\|_F^2 \quad (5)$$

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 [40] 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  $\pi = (C_1, C_2, \dots, C_k)$ , then

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

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

where  $\bar{C}_i$  is the complement of  $C_i$ , and  $\text{vol}(C_i) = \sum_{v \in C_i} d_v$ . 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

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

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

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

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

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

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

#### 2.4 Modularity Maximization

Modularity [26] 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:

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

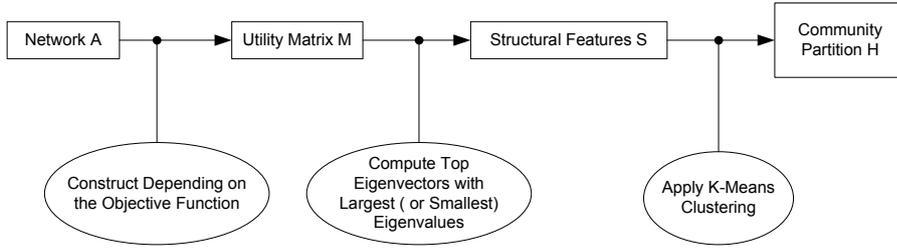


Fig. 3: A Unified View of Representative Community Detection Methods

Modularity is defined as

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

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

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

where

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

With a spectral relaxation to allow  $S$  to be continuous, the optimal  $S$  can be computed as the top- $k$  eigenvectors of matrix  $\tilde{B}$  [25] 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 Figure 3. The process is composed of 4 components with 3 intermediate steps. Given a network, a utility matrix is constructed. Depending on the objective function, different utility matrices can be constructed.

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

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.

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<sup>2</sup>. Asymptotically, the cost to construct a utility matrix is

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

Implicitly Restarted Lanczos method (IRLM) can be applied to compute the top eigenvectors efficiently [10, 42]. Let  $\ell$  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

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

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 [34] and  $\ell \ll n$ . In practice, the computation tends to be linear with respect to  $n$  if  $\ell$  is small. The post-processing step to extract community partition relies on  $k$ -means clustering, which has time complexity

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

where  $\ell$  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.

---

<sup>2</sup> The utility matrix of modularity maximization is dense but structured, thus it is rarely computed out. Its structure is exploited directly for eigenvector computation [25, 36].

### 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

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

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

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

where  $n$  is the total number of actors involved in the network. Here, we concentrate on symmetric networks<sup>3</sup>. 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 (Figure 3), we can conduct the integration in terms of each component as in Figure 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<sup>4</sup>. 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.

#### 3.1 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:

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

<sup>3</sup> Directed networks can be converted into undirected networks through certain operations as shown later.

<sup>4</sup> A preliminary work based on modularity maximization is published in [36]. This manuscript, significantly different the previous conference version, presents a general framework to interpret community detection in multi-dimensional networks.

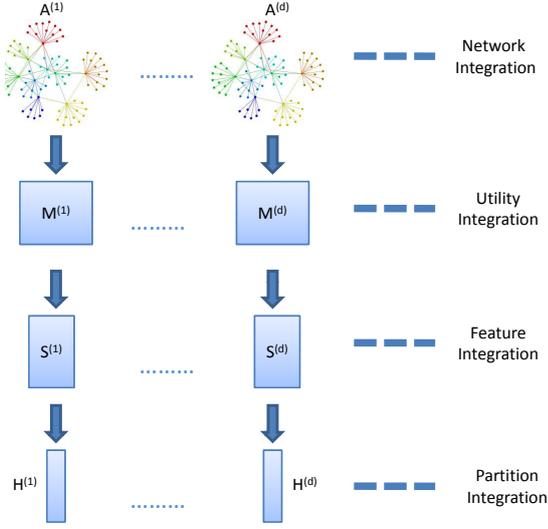


Fig. 4: Potential Multi-Dimensional Integration Strategies

Correspondingly,

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

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:

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

### 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  $i$ -th 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

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

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

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

### 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:

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

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- $\ell$  eigenvectors that maximize modularity  $Q$ , and  $V$  an orthonormal matrix such that

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

It can be verified that  $SV$  also maximizes  $Q$ :

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

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) [14, 18] 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  $i$ -th 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  $i$ -th and the  $j$ -th dimensions. Generalized CCA attempts to maximize the summation of pairwise correlations as in the following form:

$$\max \sum_{i=1}^d \sum_{j=1}^d \mathbf{w}_i^T C_{ij} \mathbf{w}_j \quad (24)$$

$$s.t. \sum_{i=1}^d \mathbf{w}_i^T C_{ii} \mathbf{w}_i = 1 \quad (25)$$

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

$$\begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1d} \\ C_{21} & C_{22} & \cdots & C_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ C_{d1} & C_{d2} & \cdots & C_{dd} \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_d \end{bmatrix} = \lambda \begin{bmatrix} C_{11} & 0 & \cdots & 0 \\ 0 & C_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & C_{dd} \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_d \end{bmatrix} \quad (26)$$

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  $diag(C_{11}, C_{22}, \dots, C_{dd})$  in Eq. (26) becomes an identity matrix. Hence  $\mathbf{w} = [\mathbf{w}_1^T, \mathbf{w}_2^T, \dots, \mathbf{w}_d^T]^T$  corresponds to the top eigenvector of the full covariance matrix on the left side of Eq. (26), which is equivalent to PCA applied to data of the following form:

$$X = [S^{(1)}, S^{(2)}, \dots, S^{(d)}] \quad (27)$$

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

$$\frac{1}{d} \sum_{i=1}^d S^{(i)} \mathbf{w}_i = \frac{1}{d} [S^{(1)}, S^{(2)}, \dots, S^{(d)}] \mathbf{w} = \frac{1}{d} X V_1 = \frac{\sigma_1}{d} U_1$$

Since  $\sigma_1/d$  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.

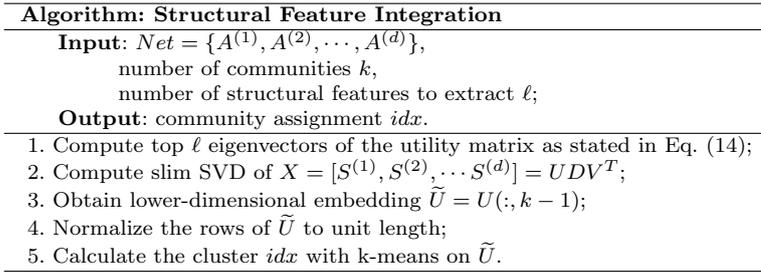


Fig. 5: Algorithm: Structural Feature Integration for Multi-Dimensional Networks

The detailed structural feature integration algorithm is summarized in Figure 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 Eq. (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 [32], which combines multiple clustering results of the same data from a variety of sources into a single consensus clustering. Strehl and Ghosh [32] 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

$$\frac{1}{d} \sum_{i=1}^d H^{(i)} (H^{(i)})^T = \frac{1}{d} \sum_{i=1}^d \tilde{H} \tilde{H}^T \quad \text{where } \tilde{H} = [H^{(1)}, H^{(2)}, \dots, H^{(d)}]$$

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  $\tilde{H}$  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 one dimension but relatively more observable in another dimension. So if there is one dimension 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) [32]. Let  $\pi^a, \pi^b$  denote two different partitions of commu-

---

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

---

Fig. 6: CDNV: Cross-Dimension Network Validation

nities. NMI is defined as

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

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

When ground truth is not available, an alternative evaluation method is needed to quantify community structures extracted employing different integration strategies. If a latent community structure is shared across network dimensions, we can perform *cross-dimension network validation* (CDNV) as in Figure 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 modularity for the data of  $A^{(p)}$  as follows:

$$Q = \frac{1}{2m} \sum_C \sum_{i \in C, j \in C} A_{ij} - d_i d_j / 2m. \quad (29)$$

A larger modularity implies more accurate community structure is discovered using the training data.

The above two evaluation methods 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.

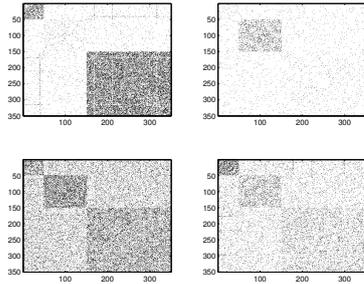


Fig. 7: One example of 4-Dimensional Network

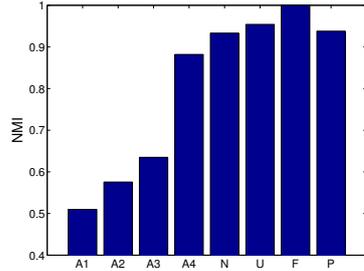


Fig. 8: Performance of different community detection methods. A1, A2, A3 and A4 show the performance on a single dimension. N, U, F, and P denote network, utility, feature, partition integration, respectively.

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<sup>5</sup> (Figure 3). Different integration strategies can be applied at various stages as shown in Figure 4. Baseline strategies are to not integrate  $(d - 1)$  dimensions, but use single dimensions.

#### 4.1 Experiments on Synthetic Data

The synthetic data has 3 groups, each having 50, 100, 200 members, respectively. There are 4 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 (Eq. (28)) can be employed to evaluate the performance of community evaluation.

<sup>5</sup> Since k-means clustering is sensitive to the initialization, we repeat k-means 5 times and pick whichever is the best as the community partition.

Table 1: Average performance over 100 runs.  $NMI$  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 cross-dimension network validation, and  $R_{CDNV}$  the ranking based on CDNV. Note that NMI and CDNV yield consistent rankings.

	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	<b>0.9351</b>	<b>1</b>	<b>0.2064</b>	<b>1</b>
	Partition	0.8048	3	0.1785	3

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 Figure 7. The first dimension of the network actually only shows 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 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 cross-dimension network validation, 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

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

Network	Dimension	Density
$A^{(1)}$	contact	$6.74 \times 10^{-4}$
$A^{(2)}$	co-contact	$1.71 \times 10^{-2}$
$A^{(3)}$	co-subscription	$4.90 \times 10^{-2}$
$A^{(4)}$	co-subscribed	$1.97 \times 10^{-2}$
$A^{(5)}$	favorite	$3.34 \times 10^{-2}$

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<sup>6</sup> is currently the most popular video sharing web site. It is reported to “attract 100 million video views per day”<sup>7</sup>. As of March 17th, 2008, there have been 78.3 million videos uploaded, with over 200, 000 videos uploaded per day<sup>8</sup>. 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<sup>9</sup> 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

<sup>6</sup> <http://www.youtube.com/>

<sup>7</sup> [http://www.usatoday.com/tech/news/2006-07-16-youtube-views\\_x.htm](http://www.usatoday.com/tech/news/2006-07-16-youtube-views_x.htm)

<sup>8</sup> <http://ksudigg.wetpaint.com/page/YouTube+Statistics?t=anon>

<sup>9</sup> <http://code.google.com/apis/youtube/overview.html>

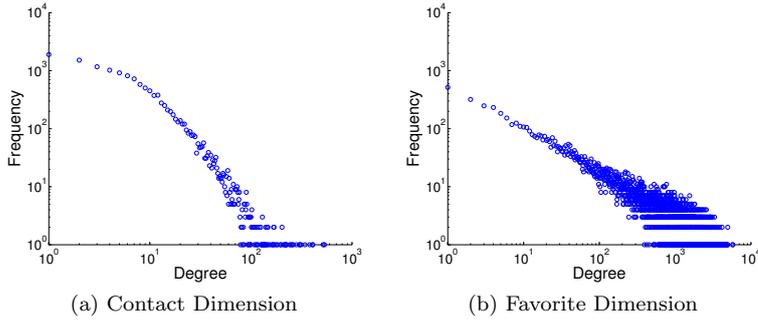


Fig. 9: Power law distribution on Different Dimensions

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:

$$A' = AA^T; \quad (30)$$

$$A'' = A^T A. \quad (31)$$

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 [19].

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;
- $A^{(2)}$  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. (30).
- $A^{(3)}$  co-subscription network: the connection between two users denotes they subscribe to the same user; constructed following Eq. (30);
- $A^{(4)}$  co-subscribed network: two users are connected if they are both subscribed by the same user; constructed following Eq. (31);
- $A^{(5)}$  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 [22], 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 share similar interests by integrating heterogeneous interactions.

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

k=20	Strategies	$R^{(1)}$	$R^{(2)}$	$R^{(3)}$	$R^{(4)}$	$R^{(5)}$	$R_{average}$
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	<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>	<b>1</b>	<b>1.00</b>
	Partition	3	3	3	3	3	3.00
k=40	Strategies	$R^{(1)}$	$R^{(2)}$	$R^{(3)}$	$R^{(4)}$	$R^{(5)}$	$R_{average}$
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	<b>1</b>	<b>1</b>	<b>1</b>	2	<b>1</b>	<b>1.20</b>
	Partition	3	3	3	<b>1</b>	3	2.60
k=60	Strategies	$R^{(1)}$	$R^{(2)}$	$R^{(3)}$	$R^{(4)}$	$R^{(5)}$	$R_{average}$
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	<b>1</b>	2	1.80
	Feature	<b>1</b>	<b>1</b>	<b>1</b>	2	<b>1</b>	<b>1.20</b>
	Partition	4	3	3	3	3	3.20

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 [8] as expected. This data set is publicly available at the first author’s homepage<sup>10</sup>.

#### 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

<sup>10</sup> [http://www.public.asu.edu/~ltang9/heterogeneous\\_network.html](http://www.public.asu.edu/~ltang9/heterogeneous_network.html)

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 cross-dimension network validation procedure, the test dimension is not available during training, thus the diagonal entries for single-dimensional methods are not shown.

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  $X$  (Eq. (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, 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.

Figure 10 and Figure 11 show the computational time of modularity maximization with respect to a variety of community numbers and network sizes.

Table 4: Time complexity of different integration strategies. *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 Eqs. (15), (16) and (17).

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	$O(ndl \cdot \min(n, dl))$	$d$	$d$	1
Partition Integration	$T_{kmeans}$	$d$	$d$	$d$

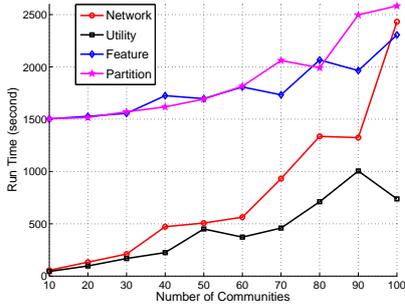


Fig. 10: Computation time with respect to varying number of communities on YouTube of 15,088 nodes

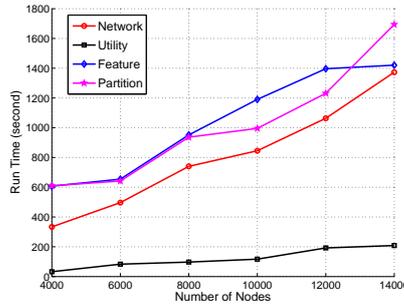


Fig. 11: Computation time with respect to varying network size (the number of communities is fixed to 40)

In both figures, feature 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 Figure 10). As shown in Figure 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

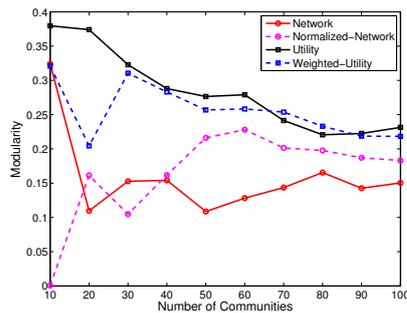


Fig. 12: Performance with Normalization (Weighting) Schemes

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:

$$\bar{A} = \sum_{i=1}^d A_{normalized}^{(i)} = \sum_{i=1}^d A^{(i)} / (2m^{(i)}) \quad (32)$$

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:

$$\bar{M} = \sum_{i=1}^d Q^{(i)} M^{(i)} \quad (33)$$

Due to the space limit, we only show the performance on the contact dimension of the YouTube network in Figure 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 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 ( $\ell$  in Figure 5). In this part, we study the performance sensitivity

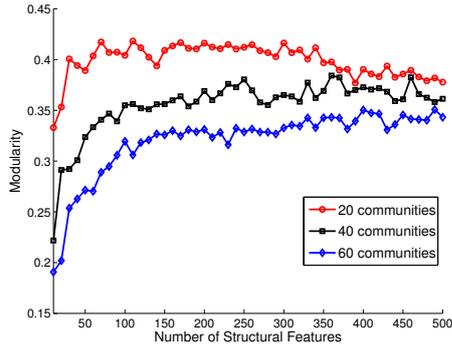


Fig. 13: Sensitivity to Number of Structural Features

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 Figure 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 cross-dimension network validation to select a proper parameter.

#### 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 Section 2.5. We can simply replace the utility matrix with network interaction or graph Laplacian as specified in Eq. (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 Figure 14. Spectral clustering is consistently better than modularity maximization and block model approximation. This result is consistent with that as reported in [42].

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

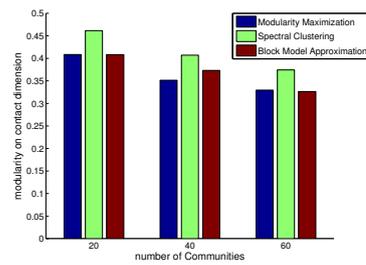


Fig. 14: Performance of Different Community Detection Methods

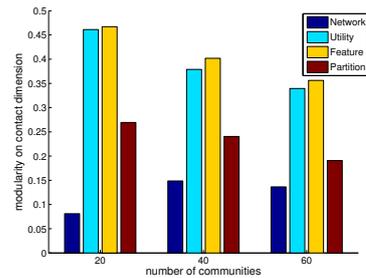


Fig. 15: Performance of Integration Schemes with Spectral Clustering

## 6 Related Work

Multi-dimensional networks (or multiple networks constructed from disparate sources) appear in many web applications. Meza et al. [1] 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 et al. [16] 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. Jin et al. [44] 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. [46] recommend documents in a digital library by integrating a citation network and networks of documents and other related entities.

Some work attempts to address unsupervised learning with multiple data sources or clustering results, such as *cluster ensemble* [32, 38, 11] and *consensus clustering* [24, 15, 27, 12]. 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 [13]. 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 [5] propose co-EM and an extension of k-means and hierarchical clustering to handle data with two conditional independent views. Sa [9] creates a bipartite based on the two views and tries to minimize the disagreement. Different spectral frameworks with multiple views are studied in [45] and [21]. 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 [37]. Similarly, [2] 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.

Unsupervised multiple kernel learning [39] is relevant to network integration if we deem each dimension of the network as a similarity or kernel matrix. Multiple kernel learning aims to find a combination of kernels to optimize for classification or clustering. Unfortunately, its limited scalability hinders its application even to a medium-size network.

Some theoretical analysis of multi-view clustering via canonical correlation analysis is presented in [7]. 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.

## 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 principal component analysis, 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 cross-dimension network validation, 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 is, in part, supported by Air Force Office of Scientific Research (AFOSR) and Office of Naval Research (ONR) grants.

## References

1. B. Aleman-Meza, M. Nagarajan, C. Ramakrishnan, L. Ding, P. Kolari, A. P. Sheth, B. I. Arpinar, A. Joshi, and T. Finin. 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*, pages 407–416, New York, NY, USA, 2006. ACM Press.
2. A. Argyriou, M. Herbster, and M. Pontil. Combining graph Laplacians for semi-supervised learning. *Advances in Neural Information Processing Systems*, 18:67, 2006.
3. L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. 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*, pages 44–54, New York, NY, USA, 2006. ACM.
4. N. Bansal, F. Chiang, N. Koudas, and F. W. Tompa. Seeking stable clusters in the blogosphere. In *VLDB '07: Proceedings of the 33rd international conference on Very large data bases*, pages 806–817, 2007.
5. S. Bickel and T. Scheffere. Multi-view clustering. In *ICDM '04: Proceedings of the Fourth IEEE International Conference on Data Mining*, pages 19–26, Washington, DC, USA, 2004. IEEE Computer Society.
6. I. Borg and P. Groenen. *Modern Multidimensional Scaling: theory and applications*. Springer, 2005.

7. K. Chaudhuri, S. M. Kakade, K. Livescu, and K. Sridharan. Multi-view clustering via canonical correlation analysis. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1–8, New York, NY, USA, 2009. ACM.
8. A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *arXiv*, 706, 2007.
9. V. R. de Sa. Spectral clustering with two views. In *Proceedings of Workshop of Learning with Multiple Views*, 2005.
10. J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst. *Templates for the solution of algebraic eigenvalue problems: a practical guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.
11. X. Z. Fern and C. E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 36, New York, NY, USA, 2004. ACM.
12. A. Goder and V. Filkov. Consensus clustering algorithms: Comparison and refinement. In *SDM*, 2008.
13. J. Hopcroft, O. Khan, B. Kulis, and B. Selman. Natural communities in large linked networks. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 541–546, New York, NY, USA, 2003. ACM.
14. H. Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936.
15. T. Hu and S. Y. Sung. Consensus clustering. *Intell. Data Anal.*, 9(6):551–565, 2005.
16. J. Jung and J. Euzenat. Towards semantic social networks. In *Proceedings of the European Semantic Web Conference (ESWC2007)*, volume 4519 of *LNCS*, pages 267–280, Berlin Heidelberg, Germany, July 2007. Springer-Verlag.
17. H. Kang, L. Getoor, and L. Singh. Visual analysis of dynamic group membership in temporal social networks. *SIGKDD Explorations, Special Issue on Visual Analytics*, 9(2):13–21, dec 2007.
18. J. Kettenring. Canonical analysis of several sets of variables. *Biometrika*, 58:433–451, 1971.
19. J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
20. D. Lizorkin, O. Medelyan, and M. Grineva. Analysis of community structure in wikipedia. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 1221–1222, New York, NY, USA, 2009. ACM.
21. B. Long, P. S. Yu, and Z. M. Zhang. A general model for multiple view unsupervised learning. In *SDM*, 2008.
22. M. McPherson, L. Smith-Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27:415–444, 2001.
23. P. Mika. Ontologies are us: A unified model of social networks and semantics. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(1):5–15, 2007. Selected Papers from the International Semantic Web Conference, International Semantic Web Conference (ISWC2005).
24. S. Monti, P. Tamayo, J. Mesirov, and T. Golub. Consensus clustering: A resampling-based method for class discovery and visualization of gene expression microarray data. *Mach. Learn.*, 52(1-2):91–118, 2003.
25. M. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)*, 74(3), 2006.
26. M. Newman. Modularity and community structure in networks. *PNAS*, 103(23):8577–8582, 2006.
27. N. Nguyen and R. Caruana. Consensus clusterings. In *ICDM '07: Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, pages 607–612, Washington, DC, USA, 2007. IEEE Computer Society.
28. G. Palla, A.-L. Barabasi, and T. Vicsek. Quantifying social group evolution. *Nature*, 446(7136):664–667, April 2007.
29. M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *KDD*, pages 61–70, 2002.
30. P. Sarkar and A. W. Moore. Dynamic social network analysis using latent space models. *SIGKDD Explor. Newsl.*, 7(2):31–40, 2005.

31. L. Specia and E. Motta. Integrating folksonomies with the semantic web. In *Proceedings of the 4th European conference on The Semantic Web: Research and Applications*, pages 624–639, Berlin, Heidelberg, 2007. Springer-Verlag.
32. A. Strehl and J. Ghosh. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617, 2003.
33. L. Tang and H. Liu. Relational learning via latent social dimensions. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 817–826, New York, NY, USA, 2009. ACM.
34. L. Tang and H. Liu. Scalable learning of collective behavior based on sparse social dimensions. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1107–1116, New York, NY, USA, 2009. ACM.
35. L. Tang, H. Liu, J. Zhang, and Z. Nazeri. Community evolution in dynamic multi-mode networks. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 677–685, New York, NY, USA, 2008. ACM.
36. L. Tang, X. Wang, and H. Liu. Uncovering groups via heterogeneous interaction analysis. In *Proceeding of IEEE International Conference on Data Mining*, pages 503–512, 2009.
37. W. Tang, Z. Lu, and I. S. Dhillon. Clustering with multiple graphs. In *ICDM '09: Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, pages 1016–1021, Washington, DC, USA, 2009. IEEE Computer Society.
38. A. Topchy, A. K. Jain, and W. Punch. Combining multiple weak clusterings. In *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, page 331, Washington, DC, USA, 2003. IEEE Computer Society.
39. H. Valizadegan and R. Jin. Generalized maximum margin clustering and unsupervised kernel learning. In *NIPS*, 2007.
40. U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.
41. S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.
42. S. White and P. Smyth. A spectral clustering approaches to finding communities in graphs. In *SDM*, 2005.
43. I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufman, 2005.
44. Y. M. Yingzi Jin and M. Ishizuka. 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*, New York, NY, USA, April 21-25 2008. ACM Press.
45. D. Zhou and C. J. C. Burges. Spectral clustering and transductive learning with multiple views. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 1159–1166, New York, NY, USA, 2007. ACM.
46. D. Zhou, S. Zhu, K. Yu, X. Song, B. L. Tseng, H. Zha, and C. L. Giles. Learning multiple graphs for document recommendations. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 141–150, New York, NY, USA, 2008. ACM.