Community Detection with Partially Observable Links and Node Attributes

Xiaokai Wei*, Bokai Cao*, Weixiang Shao†, Chun-Ta Lu* and Philip S. Yu*
*Department of Computer Science
University of Illinois at Chicago, Chicago, IL
Email: {xwei2, caobokai, clu29, psyu}@uic.edu
† Google Inc., Mountain View, CA
Email: software.shao@gmail.com

***Abstract*—Community detection has been an important task for social and information networks. Existing approaches usually assume the completeness of linkage and content information. However, the links and node attributes can usually be partially observable in many real-world networks. For example, users can specify their privacy settings to prevent non-friends from viewing their posts or connections. Such incompleteness poses additional challenges to community detection algorithms. In this paper, we aim to detect communities with partially observable link structure and node attributes. To fuse such incomplete information, we learn link-based and attribute-based representations via kernel alignment and a co-regularization approach is proposed to combine the information from both sources (i.e., links and attributes). The link-based and attribute-based representations can lend strength to each other via the partial consensus learning. We present two instantiations of this framework by enforcing hard and soft consensus constraint respectively. Experimental results on real-world datasets show the superiority of the proposed approaches over the baseline methods and its robustness under different observable levels.**

I. Introduction

Recent years have witnessed an ever-growing number of information networks in a wide range of domains. Examples include computer networks, biological, semantic and social networks. In these networks, feature vectors are usually associated with nodes [23] [22] and links represent relationships between nodes. Community detection [12], [28], which aims at grouping similar objects into one community while partitioning dissimilar objects into different communities, is an important step to understand the characteristics of these networks. By identifying network communities, we are able to study the interactions between social groups [1], infer missing attribute values of nodes [2], [6], predict potential connections [5] between nodes, and identify nodes with similar functionalities [18].

Conventional community detection algorithms for network data are mainly designed for exploiting linkage information alone without taking advantage of the content information [20] [25]. However, in the era of big data, one can often collect node attributes, such as user profiles in social networks, properties of genes in biological networks and paper titles/abstracts in co-authorship networks. These node attributes can also provide important information for characterizing different communities. Therefore, it is critical to use the complementary vector-based node attributes in community detection tasks, especially to alleviate the *cold start* problem [15] [26]. For example, considering a new user joining a social network with few friendship/following connections (i.e., links), we must leverage his/her user profile (i.e., node attributes) fulfilled in the registration process to infer the correct community affiliation. Figure 1 illustrates this example with six nodes in the network. Although node v_6 has no connection in the link structure view, we should be able to affiliate it with nodes v_2 and v_4 in the node attribute view, since they are close to node v_6 in the feature space.

On the other hand, in real-world networks, attributes of a node can be totally unobservable due to a variety of reasons. For example, a prudent user may set a very strict privacy level with which no one or only friends could view his/her profile and posts. In this scenario, the communities of such nodes can still be inferred by their link structure. Consider the example in Figure 1, the vector-based feature representation of node v_5 is missing in the node attribute view, while we could observe that the linkage structure of nodes v_1, v_3 and v_5 tightly forms a triangle which indicates a potential community.

Therefore, to handle increasingly complicated settings in this big data era, an effective and robust community detection method should blend the two complementary (and potentially incomplete) information sources (i.e., links and attributes) into a unified framework. Although some recent works [26], [14] have proposed to detect communities based on both the network structure and node content, they usually assume that both sources of information are fully observable. However, such assumption may not hold in practice. In fact, the link structure and node attributes are often partially observable in diverse scenarios:

- In popular real-world social networks (e.g., Twitter, Facebook and LinkedIn), new users usually have no connection, and some users may specify a strict privacy setting so that their connections, personal information or posts cannot be viewed by non-friends[1][2]. Identifying users in the same social group or with similar interests

[1]https://www.facebook.com/help/325807937506242/
[2]https://help.linkedin.com/app/answers/detail/a_id/52

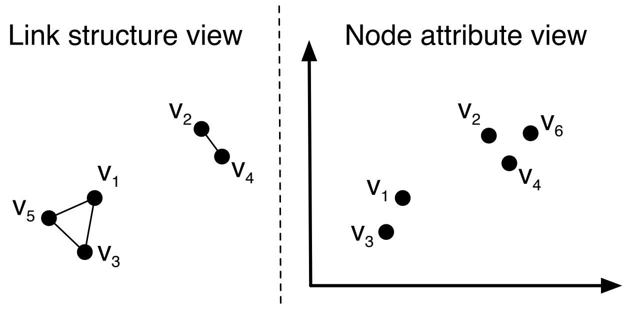


Figure 1: An example of partially observable networks

in such partially observable setting could facilitate better personalized services.

- In bioinformatics information networks, discovering genes in the same gene family could help researchers understand the complex interactions between genes. However, it is difficult to collect all the experimental data for all the possible pairs of genes. Moreover, it can be costly to obtain features for certain genes, such as the newly discovered ones.
- In terrorist-attack networks where nodes represent terrorist activities and links represent terrorist attacks that are committed by the same terrorist organization. Detecting communities in these networks is crucial for analyzing the underlying organization of terrorist-attack activities. However, investigating the community structures within these networks is challenging, since the complete linkages between attacks are not clearly resolved [11]. It is even more difficult to obtain all the attributes for a surprise attack.

In the partially observable networks, one can observe only a small fraction of nodes with both connections and node attributes, and the other nodes have either links or attributes unobservable. While it is possible that certain nodes have neither links nor attributes disclosed, we do not consider such nodes since no information can be exploited to infer their community affiliations.

In this paper, we propose a novel approach, POLNA (community detection for Partially Observable Links and Node Attributes), to combine such incomplete sources of information. In the original form, nodes with only observable links and nodes with only observable attributes are not directly comparable, which poses challenges for deciding their community affiliations. To address this challenge, we learn a latent representation for each node via kernel alignment based on either link structure or node content. The link-based representations and content-based representations of fully observable nodes interact with each other by co-regularization. Each node is equipped with representation in a latent space and hence comparable to each other. Experimental results on real-world datasets show that POLNA outperforms baseline methods, especially when only a small number of nodes are fully observable. The contribution of the paper can be summarized as follows:

- We identify the problem of community detection for partially observable networks, which widely exist in many applications. To our best knowledge, this is the first attempt to address the challenges of identifying communities on networks with both incomplete links and node attributes.
- We propose a latent representation-based approach to fuse information from incomplete sources (i.e., partially observable links and attributes).
- Two instantiations (soft and hard consensus constraint) of the proposed framework POLNA are developed and their relative strengths are discussed.

The rest of the paper is organized as follows. In section II, we briefly review related work on community detection. In section III, we provide some definitions and background knowledge for our framework. We present the co-regularized framework with two variants in section IV and discuss the optimization methods in section V. Experimental results are shown in section VI and we conclude our work in section VII.

II. Related Work

Existing community detection methods can be roughly categorized into two classes: approaches that utilize only links [12], [9], [25] and approaches that utilize both links and node content [27], [28], [26]. Newman et al. introduce *Modularity* to measure the graph partitioning quality [12]. Karrer et al. propose a degree-corrected Stochastic Block Model for community detection [9]. Yang et al. [25] and Wang et al. [20] use Non-negative Matrix Factorization (NMF) to detect communities based on link structure. Yang et al. [26] and Pei et al. [14] further discuss how to incorporate node attributes into the NMF-based framework for community detection.

Community detection methods can also be classified by whether they aim to detect disjoint [12] or overlapping communities [26], [25]. Disjoint community detection methods assign each node a hard community membership, while overlapping community detection methods assume each node can belong to multiple communities. For example, some probabilistic models [3], [24] assume mixture models (i.e., probabilities of community affiliations sum to one) for soft community affiliation. BigCLAM [25] and CESNA [26] further relax the simplex constraint on community membership and allow a node to have high membership strength in multiple communities simultaneously. Our method is designed for non-overlapping community detection.

All these methods assume that the node information (links or attributes) are fully observable, except DSHRINK [11] which studies community detection with incomplete linkage information. However, it still assumes the completeness of node attributes, which is different from our setting. To our best knowledge, POLNA (proposed in this paper) is the first method that attempts to address the scenario where both links and attributes can be partially observable.

III. Preliminaries

In this section, we present several concepts as preliminaries that will be used throughout this paper.

Definition 1: **Information Network** An information network $G = (V, E, X)$ consists of V, the set of nodes, $E \subseteq V \times V$, the set of links, and the feature matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$ ($n = |V|$), where $\mathbf{x}_i \in \mathbb{R}^D$ ($i = 1 \dots n$) is the feature vector of node v_i.

Definition 2: **Partially Observable Information Network** In a partially observable information network $G = (V, E, X)$, each node belongs to one or two (overlapping) of the following sets: the set of nodes with observable links (denoted as O^g) and nodes with observable attributes (denoted as O^a). We further denote the set of nodes with both observable links and attributes as $O^s = O^g \cap O^a$. Note that $V = O^g \cup O^a$ since we assume each node has at least one source of information.

Let the number of nodes with observable links or observable attributes be $n^g = |O^g|$ and $n^a = |O^a|$, respectively. For more concise notations in the following sections, we assign local indices $1 \sim n^g$ to the nodes in O^g and $1 \sim n^a$ to the nodes in O^a. A mapping function $\phi^g(i)$ (or $\phi^a(i)$) maps the local index $i \in \{1, \dots, n^g\}$ (or $i \in \{1, \dots, n^a\}$) in O^g (or O^a) to its global index. A summary of symbols used in this paper is shown in Table I.

Table I: Symbol definitions

Symbol	Definition
$\mathbf{x}_i \in \mathcal{R}^D$	Feature vector of the i-th data sample
$\Phi(\cdot)$	Feature mapping function
$\phi^g(\cdot)$ / $\phi^a(\cdot)$	Index mapping function for link-observable/attribute-observable nodes from local index to global index
$\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$	Centering matrix
K	Kernel function
K_c	Centered version of kernel function
$\mathbf{L}^g$	Relation matrix of link-observable nodes
$\mathbf{L}^a$	Kernel matrix over original features of attribute-observable nodes
O^g / O^a	Set of link-observable/attribute-observable nodes
O^s	Set of nodes with both observable link and observable attributes
$\mathbf{U}^g$ / $\mathbf{U}^a$	Latent representation of link-observable/attribute-observable nodes
$\mathbf{U}^*$	Consensus latent representation
$\lVert \cdot \rVert_F$	Frobenius norm of matrix $(\cdot)$

Next, we briefly review centered kernel and kernel alignment following [7] [21]. A feature mapping $\Phi(\mathbf{x}) : \mathcal{X} \to \mathcal{F}$ can be centered by subtracting its expectation, $\Phi(\mathbf{x}) - E[\Phi(\mathbf{x})]$. Centering a kernel function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ consists of centering all the feature mapping $\Phi(\cdot)$ associated with K.

Definition 3: **Centered Kernel Function** The kernel function $K \in \mathbb{R}^{n\times n}$ after centering is

$$K_c(\mathbf{x}_i, \mathbf{x}_j) = (\Phi(\mathbf{x}_i) - E[\Phi(\mathbf{x}_i)])^T(\Phi(\mathbf{x}_j) - E[\Phi(\mathbf{x}_j)]) \tag{1}$$

For a finite number of samples, we can have similar definitions. A feature vector is centered by subtracting its empirical expectation, $\Phi(\mathbf{x}_i) - \bar{\Phi}$, where $\bar{\Phi} = \frac{1}{n}\sum_{j=1}^{n}\Phi(\mathbf{x}_j)$.

Definition 4: **Centered Kernel Matrix** The kernel matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$ after centering is

$$(\mathbf{K}_c)_{ij} = K_{ij} - \frac{1}{n}\sum_{i=1}^{n} K_{ij} - \frac{1}{n}\sum_{j=1}^{n} K_{ij} + \frac{1}{n^2}\sum_{i,j=1}^{n} K_{ij} \tag{2}$$

By denoting the centering matrix $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$, the kernel matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$ after centering can be expressed as $\mathbf{K}_c = \mathbf{HKH}$.

Similar to calculating the cosine similarity between two vectors, one can compute the similarity between two matrices as follows, which is usually referred to as *Kernel Matrix Alignment*.

Definition 5: **Kernel Matrix Alignment** For two kernel matrices $\mathbf{K}_1 \in \mathbb{R}^{n\times n}$ and $\mathbf{K}_2 \in \mathbb{R}^{n\times n}$ (assume $||\mathbf{K}_1||_F > 0$ and $||\mathbf{K}_2||_F > 0$), the alignment between $\mathbf{K}_1$ and $\mathbf{K}_2$ is defined as

$$\rho(\mathbf{K}_1, \mathbf{K}_2) = \frac{\mathrm{Tr}(\mathbf{K}_1\mathbf{K}_2)}{||\mathbf{K}_1||_F \cdot ||\mathbf{K}_2||_F} \tag{3}$$

where $\mathrm{Tr}(\cdot)$ is the trace of a matrix.

Intuitively, this measure can be viewed as calculating the angle/cosine similarity between two vectorized kernel matrices. In many tasks, the normalization term $||\mathbf{K}_1||_F \cdot ||\mathbf{K}_2||_F$ usually makes the optimization problem more difficult to solve, so it is usually desirable to use the unnormalized version for easier optimization.

Definition 6: **Unnormalized Kernel Alignment** For two kernel matrices $\mathbf{K}_1 \in \mathbb{R}^{n\times n}$ and $\mathbf{K}_2 \in \mathbb{R}^{n\times n}$, where

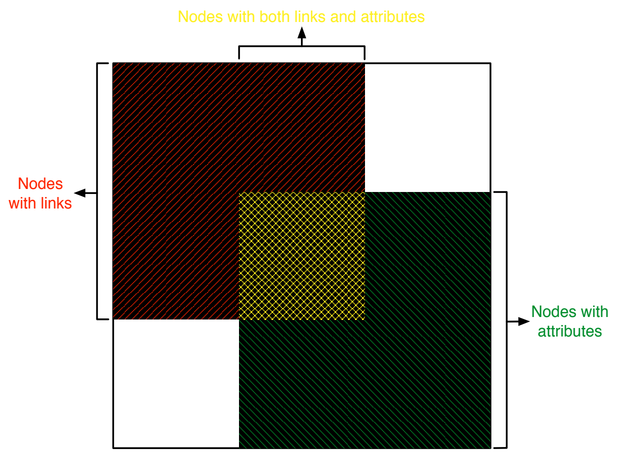


Figure 2: Partially aligned attribute matrix and link matrix

$||\mathbf{K}_1||_F > 0$ and $||\mathbf{K}_2||_F > 0$, the alignment between $\mathbf{K}_1$ and $\mathbf{K}_2$ is defined as

$$\rho(\mathbf{K}_1, \mathbf{K}_2) = \mathrm{Tr}(\mathbf{K}_1\mathbf{K}_2) \quad (4)$$

Although such alignment technique is usually applied to kernel matrices, it can also be used on other symmetric real matrices, such as the undirected networks, since the matrix similarity interpretation still holds.

IV. Proposed Method

In this section, we describe the framework of POLNA based on matrix alignment and develop two variants of POLNA with hard and soft constraints, respectively.

A. Representation Learning by Kernel Alignment

The key idea of our proposed method is to learn latent representations that best align with the network structure and attribute kernel. For the network nodes, we aim to learn a link-based representation $\mathbf{U}^g \in \mathbb{R}^{n^g \times c}$ (for nodes in O^g) and attribute-based representation $\mathbf{U}^a \in \mathbb{R}^{n^a \times c}$ (for nodes in O^a) by kernel matrix alignment, where c is the latent dimension size. Suppose $\mathbf{K}^g \in \mathbb{R}^{n^g \times n^g}$ and $\mathbf{K}^a \in \mathbb{R}^{n^a \times n^a}$ are kernel matrices computed from the latent representation $\mathbf{U}^g$ and $\mathbf{U}^a$ with Gaussian Kernel, respectively.

$$\begin{aligned} K^g_{ij} &= e^{-||\mathbf{U}^g_i - \mathbf{U}^g_j||^2} \\ K^a_{ij} &= e^{-||\mathbf{U}^a_i - \mathbf{U}^a_j||^2} \end{aligned} \quad (5)$$

where $\mathbf{U}^g_i \in \mathbb{R}^c$ $(i = 1, 2, \dots, n^g)$ and $\mathbf{U}^a_i \in \mathbb{R}^c$ $(i = 1, 2, \dots, n^a)$ are the link-based and attribute-based representations for local index i, respectively.

For the nodes with observable attributes (i.e., O^a), we construct a kernel matrix $\mathbf{L}^a \in \mathbb{R}^{n^a \times n^a}$ from the node attributes. Let $\mathbf{L}^a_c$ and $\mathbf{K}^a_c$ denote the centered version of $\mathbf{L}^a$ and $\mathbf{K}^a$, respectively. We learn the latent representation by maximizing the following (centered) kernel function alignment:

$$\mathrm{Tr}(\mathbf{L}^a_c\mathbf{K}^a_c) = \mathrm{Tr}(\mathbf{H}\mathbf{L}^a\mathbf{H}\mathbf{H}\mathbf{K}^a\mathbf{H}) = \mathrm{Tr}(\mathbf{H}\mathbf{L}^a\mathbf{H}\mathbf{K}^a) \quad (6)$$

where the second equation can be obtained by the fact that $\mathbf{H}\mathbf{H} = \mathbf{H}$ and the trace property $\mathrm{Tr}(\mathbf{A}\mathbf{B}) = \mathrm{Tr}(\mathbf{B}\mathbf{A})$, for all matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n^a \times n^a}$. To learn the attribute-based representation, we minimize the following objective.

$$\min_{\mathbf{U}^a} f^a = -\mathrm{Tr}(\mathbf{H}\mathbf{L}^a\mathbf{H}\mathbf{K}^a) + \lambda||\mathbf{U}^a||^2_F \quad (7)$$

where λ controls the complexity of the latent factors. By minimizing the objective function over $\mathbf{U}^a$, we can obtain a latent representation of nodes that behaves similarly as the original attribute kernel. Such a criterion is also known as HSIC [8], and CLUHSIC [17] uses a similar objective for clustering purpose. But CLUHSIC seeks to find '0/1' cluster membership assignments in a greedy manner, while our approach uses gradient-based method to get continuous-value latent factors.

Similarly, to learn link-based representations $\mathbf{U}^g$ from network structure, we can minimize the following objective

$$\min_{\mathbf{U}^g} f^g = -\mathrm{Tr}(\mathbf{H}\mathbf{L}^g\mathbf{H}\mathbf{K}^g) + \lambda||\mathbf{U}^g||^2_F \quad (8)$$

where $\mathbf{L}^g \in \{0,1\}^{n^g \times n^g}$ represents the links between the link-observable nodes. Although the partial network matrix $\mathbf{L}^g$, strictly speaking, is not a kernel matrix, such an objective can be interpreted as the similarity of two matrices and we found it performs very well empirically.

For the partial kernel matrix $\mathbf{L}^a$ over original features, we found using an all-pair kernel matrix usually underperforms using a kNN matrix. This is perhaps because the distance computed on non-nearest pairs is not as reliable as the distance computed on nearest-neighbor pairs, since the original attribute space tend to be noisy. In spectral clustering [13] or graph regularization [4], using the kNN graph rather than the full graph is also a common practice. Therefore, we use a kNN matrix with '0-1' weighting as $\mathbf{L}^a$ and only retain the pairs in which one sample is within the other sample's k nearest neighbors in the matrix.

B. Partial Co-regularization

Links and node attributes provide complementary information on the nodes. So it is desirable to learn a consensus from the link-based representation and attribute-based representation.

Now we discuss how to instantiate the consensus learning. Let us denote $f^{g*} = -\mathrm{Tr}(\mathbf{H}\mathbf{L}^g\mathbf{H}\mathbf{K}^g)$ and $f^{a*} = -\mathrm{Tr}(\mathbf{H}\mathbf{L}^a\mathbf{H}\mathbf{K}^a)$. One simple approach is to enforce a strict consensus which requires $\mathbf{U}^g_{\phi^{g^{-1}}(i)} = \mathbf{U}^a_{\phi^{a^{-1}}(i)} = \mathbf{U}^*_i$

$(\forall i = 1, \ldots, n)$.

$$\begin{aligned}\min_{\mathbf{U}^a, \mathbf{U}^g, \mathbf{U}^*} f =& f^{g*} + f^{a*} + f^{\lambda *} \\ =& -\operatorname{Tr}(\mathbf{H}\mathbf{L}^g\mathbf{H}\mathbf{K}^g) \\ & -\operatorname{Tr}(\mathbf{H}\mathbf{L}^a\mathbf{H}\mathbf{K}^a) \\ & + \lambda ||\mathbf{U}^*||_F^2 \\ \text{s.t. } & \mathbf{U}^g_{\phi^{g-1}(i)} = \mathbf{U}^a_{\phi^{a-1}(i)} = \mathbf{U}^*_i\end{aligned} \tag{9}$$

where $\phi^{g^{-1}}(i)$ and $\phi^{a^{-1}}(i)$ map the global index i to the corresponding local indices. We refer to this formulation with hard constraint as POLNA-HC. Since $\mathbf{U}^g_{\phi^{g-1}(i)} = \mathbf{U}^a_{\phi^{a-1}(i)} = \mathbf{U}^*_i$, we can write Eq (9) in a more succinct form with only $\mathbf{U}^*$:

$$\begin{aligned}\min_{\mathbf{U}^*} f =& -\operatorname{Tr}(\mathbf{H}\mathbf{L}^g\mathbf{H}\mathbf{K}^g) \\ & -\operatorname{Tr}(\mathbf{H}\mathbf{L}^a\mathbf{H}\mathbf{K}^a) \\ & + \lambda ||\mathbf{U}^*||_F^2\end{aligned} \tag{10}$$

Enforcing the link representation and attribute representation to be exactly the same might be too strict, since links and attributes could contain certain information reflecting their unique characteristics. A more flexible option is to impose a soft constraint on $\mathbf{U}^a$ and $\mathbf{U}^g$ by adopting a co-regularization approach:

$$\min_{\mathbf{U}^*} f^s = \sum_{i \in O^s} (||\mathbf{U}^g_{\phi^{g-1}(i)} - \mathbf{U}^*_i||_F^2 + ||\mathbf{U}^a_{\phi^{a-1}(i)} - \mathbf{U}^*_i||_F^2) \tag{11}$$

Hence, the total objective function is as follows.

$$\begin{aligned}\min_{\mathbf{U}^a, \mathbf{U}^g, \mathbf{U}^*} f =& f^g + f^a + f^s \\ =& -\operatorname{Tr}(\mathbf{H}\mathbf{L}^g\mathbf{H}\mathbf{K}^g) + \lambda ||\mathbf{U}^g||_F^2 \\ & -\operatorname{Tr}(\mathbf{H}\mathbf{L}^a\mathbf{H}\mathbf{K}^a) + \lambda ||\mathbf{U}^a||_F^2 \\ & + \alpha \sum_{i \in O^s} (||\mathbf{U}^g_{\phi^{g-1}(i)} - \mathbf{U}^*_i||_F^2 + ||\mathbf{U}^a_{\phi^{a-1}(i)} - \mathbf{U}^*_i||_F^2)\end{aligned} \tag{12}$$

where f^a and f^g are defined in Eq (7) and Eq (8), and α controls the penalty on enforcing the consensus. We refer to this soft-constraint version of POLNA as POLNA-SC. It is easy to observe that POLNA-HC can be viewed as a special case of POLNA-SC where $\alpha = +\infty$.

C. Discussion

In partially observable networks, consider the following two types of nodes (Figure 2):

Nodes with both links and attributes Although we assume that many nodes only have either links or attributes observable, there are still certain nodes (i.e., O^s) with both information sources. To exploit such fully observable nodes with their link-based and attribute-based representations, we propose to learn a consensus $\mathbf{U}^*$ on these nodes. Since these nodes utilize both sources of information, one can expect their representation to be of high quality.

Nodes with either links or attributes With the consensus learning, we learn the latent representation for nodes in O^s by incorporating information from both sources. For the nodes with only one source of information, they can still benefit from the interaction with the nodes in O^s, as the representations of nodes in O^s have good quality.

To summarize, with either hard or soft constraint, the fully observable nodes serve as bridges so that the link-based representation and attribute-based representation can lend strength to each other. Hence, the latent representation of nodes with only one type of information can also be enhanced.

V. Optimization

In this section, we discuss how to solve the optimization problem with hard or soft consensus constraint.

Now we derive the gradient for optimizing the objective function in Eq (9) and Eq (12). We only show the results w.r.t $\mathbf{U}^g$, and gradient w.r.t $\mathbf{U}^a$ can be derived in a similar manner. For Gaussian kernel, we can derive the following gradient:

$$\frac{\partial K^g_{ij}}{\partial \mathbf{U}^g_i} = -K^g_{ij} \cdot 2 \left(\mathbf{U}^g_i - \mathbf{U}^g_j\right) \tag{13}$$

Note that the indices of $\mathbf{U}^g$ here are the local indices.

By noting $\operatorname{Tr}(\mathbf{AB}) = \sum_{i,j} (\mathbf{A} \odot \mathbf{B})_{ij}$ for symmetric $\mathbf{B}$ ($\odot$ is the element-wise product), we can obtain the following gradient on f^g:

$$\begin{aligned}\frac{\partial f^g}{\partial \mathbf{U}^g_i} &= -\sum_{j=1}^{n^g} ((\mathbf{H}\mathbf{L}^g\mathbf{H})_{ij} \cdot \frac{\partial K^g_{ij}}{\partial \mathbf{U}^g_i} + (\mathbf{H}\mathbf{L}^g\mathbf{H})_{ji} \cdot \frac{\partial K^g_{ji}}{\partial \mathbf{U}^g_i}) + 2\lambda \mathbf{U}^g_i \\ &= 4 \cdot \sum_{j=1}^{n^g} (((\mathbf{H}\mathbf{L}^g\mathbf{H}) \odot \mathbf{K}^g)_{ij} \left(\mathbf{U}^g_i - \mathbf{U}^g_j\right)) + 2\lambda \mathbf{U}^g_i\end{aligned} \tag{14}$$

A. Optimization with Hard Constraint

In POLNA-HC, we enforce link-based representation $\mathbf{U}^g$ and attribute-based representation $\mathbf{U}^a$ to be the same (i.e., $\mathbf{U}^* = \mathbf{U}^g = \mathbf{U}^a$). So we can derive the following gradient w.r.t $\mathbf{U}^*$.

$$\frac{\partial f}{\partial \mathbf{U}^*_i} = \begin{cases} \frac{\partial f^{g*}}{\partial \mathbf{U}^g_{\phi^{g-1}(i)}} + 2\lambda \mathbf{U}^*_i & \text{for } \phi(i) \in O^g \backslash O^s \\ \frac{\partial f^{g*}}{\partial \mathbf{U}^a_{\phi^{g-1}(i)}} + 2\lambda \mathbf{U}^*_i & \text{for } \phi(i) \in O^a \backslash O^s \\ \frac{\partial f^{g*}}{\partial \mathbf{U}^g_{\phi^{g-1}(i)}} + \frac{\partial f^{a*}}{\partial \mathbf{U}^a_{\phi^{a-1}(i)}} + 2\lambda \mathbf{U}^*_i & \text{for } \phi(i) \in O^s \end{cases} \tag{15}$$

In Eq (15), if node i has both links and attributes, $\mathbf{U}^*_i$ needs to be updated with gradients derived from both link kernel alignment and attribute kernel alignment; if node i only has link information (or attribute information), it just needs to be updated by the gradient from corresponding type of matrix alignment.

Algorithm 1 Optimization for POLNA with Hard Consensus Constraint (POLNA-HC)

```
Input: (Partially observable) feature matrix $\mathbf{X}$, (Partially observable) network $G$, $\lambda$.
Output: Consensus representation $\mathbf{U}^*$
1: Initialize: $\mathbf{U}_i^g = rand(0,1)$, $\mathbf{U}_i^a = rand(0,1)$, $\mathbf{U}^* = \mathbf{0}$
2: Update $\mathbf{U}^*$ with Eq (15) by L-BFGS
```

Algorithm 2 Alternating Optimization for POLNA with Soft Consensus Constraint (POLNA-SC)

```
Input: (Partially observable) feature matrix $\mathbf{X}$, (Partially observable) network $G$, $\lambda$ and $\alpha$.
Output: Consensus representation $\mathbf{U}^*$.
1: Initialize: $\mathbf{U}_i^g = rand(0,1)$, $\mathbf{U}_i^a = rand(0,1)$, $\mathbf{U}^* = \mathbf{0}$
2: Set $t = 1$
3: while not converged do
4:    Find the optimal $\mathbf{U}^g$ by L-BFGS with Eq (16)
5:    if $t = 1$ then
6:       $\mathbf{U}_i^* = \mathbf{U}^g_{\phi^{g-1}(i)}, \forall i \in O^s$
7:    end if
8:    Find the optimal $\mathbf{U}^a$ by L-BFGS with Eq (17)
9:    $\mathbf{U}_i^* = (\mathbf{U}^g_{\phi^{g-1}(i)} + \mathbf{U}^a_{\phi^{a-1}(i)})/2, \quad \forall i \in O^s$
10:   $t = t + 1$
11: end while
12: $\mathbf{U}_i^* = \mathbf{U}^g_{\phi^{g-1}(i)}, \quad \forall i \in O^g/O^s$
13: $\mathbf{U}_i^* = \mathbf{U}^a_{\phi^{a-1}(i)}, \quad \forall i \in O^a/O^s$
```

B. Optimization with Soft Constraint

For POLNA-SC, we need to optimize over $\mathbf{U}^*$, $\mathbf{U}^g$ and $\mathbf{U}^a$. In this subsection, we develop an alternating optimization approach to solve the problem in Eq (12).

Step 1. Fixing $\mathbf{U}^*$, update $\mathbf{U}^g$ and $\mathbf{U}^a$.

To sum up, the gradient of the objective function in Eq (12) w.r.t $\mathbf{U}_i^g$ is as follows:

$$\frac{\partial f}{\partial \mathbf{U}_i^g} = \begin{cases} \frac{\partial f^g}{\partial \mathbf{U}_i^g} & \text{for } \phi(i) \in O^g \backslash O^s \\ \frac{\partial f^{\hat{g}}}{\partial \mathbf{U}_i^g} + \frac{\partial f^s}{\partial \mathbf{U}_i^g} & \text{for } \phi(i) \in O^s \end{cases} \tag{16}$$

Similarly, one can derive the following for $\mathbf{U}_i^a$:

$$\frac{\partial f}{\partial \mathbf{U}_i^a} = \begin{cases} \frac{\partial f^a}{\partial \mathbf{U}_i^a} & \text{for } \phi(i) \in O^a \backslash O^s \\ \frac{\partial f^{\hat{a}}}{\partial \mathbf{U}_i^a} + \frac{\partial f^s}{\partial \mathbf{U}_i^a} & \text{for } \phi(i) \in O^s \end{cases} \tag{17}$$

We can also derive the following gradient on f^s w.r.t $\mathbf{U}_i^g$:

$$\frac{\partial f^s}{\partial \mathbf{U}_i^g} = 2\alpha(\mathbf{U}_i^g - \mathbf{U}^*_{\phi^g(i)}) \tag{18}$$

We can use gradient-based method (e.g., steepest descent or L-BFGS) to solve this subproblem.

Table II: Statistics of two datasets

Statistics	Citeseer	Cora
# of instances	3312	2708
# of links	4598	5429
# of attributes	3703	1433
# of classes	6	7

Step 2. Fixing $\mathbf{U}^g$ and $\mathbf{U}^a$, update $\mathbf{U}^*$. With fixed $\mathbf{U}^g$ and $\mathbf{U}^a$, we find the optimal $\mathbf{U}^*$ for the following problem.

$$\min_{\mathbf{U}^*} f^s = \sum_{i \in O^s} ||\mathbf{U}^g_{\phi^{g-1}(i)} - \mathbf{U}_i^*||_F^2 + ||\mathbf{U}^a_{\phi^{a-1}(i)} - \mathbf{U}_i^*||_F^2 \tag{19}$$

It is easy to verify that the optimal $\mathbf{U}_i^*$ for Eq (19) has a closed form solution $\mathbf{U}_i^* = (\mathbf{U}^g_{\phi^{g-1}(i)} + \mathbf{U}^a_{\phi^{a-1}(i)})/2$.

After algorithm converges, $\mathbf{U}_i^*$ of nodes $i \in O_s$ are directly available. For the nodes $i \in O^g/O^s$ (or $i \in O^a/O^s$), we populate them with corresponding values from $\mathbf{U}^g$ (or $\mathbf{U}^a$). We then apply kMeans to $\mathbf{U}^*$ to derive disjoint community memberships for the nodes. Algorithm 1 and Algorithm 2 summarize the optimization methods for POLNA-HC and POLNA-SC, respectively. POLNA-SC offers more flexibility by the soft constraint while POLNA-HC can be more efficient since it does not require alternating optimization.

Convergence proof for POLNA-SC: The alternating optimization framework in Algorithm 2 is essentially blockwise coordinate descent [19] with three blocks of variables: $\mathbf{U}^a$, $\mathbf{U}^g$ and $\mathbf{U}^*$. So convergence can be guaranteed based on the general proof of convergence for blockwise coordinate descent.

VI. Experiments

In this section, we perform community detection on real-world networks with partially observable links and node attributes.

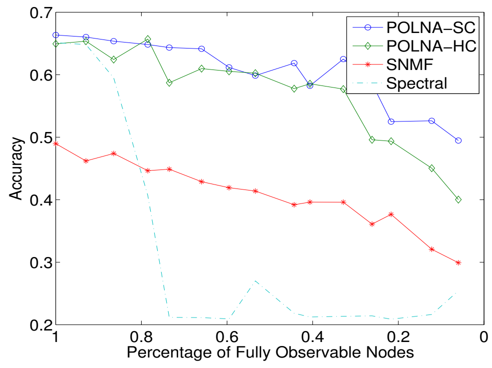
A. Datasets

We use two publicly available network datasets[3] [16] in our experiments.

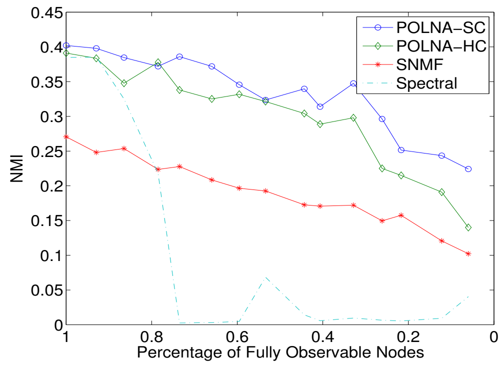
- Citeseer Dataset[3]: A citation network in computer science divided into six subareas (Agents, AI, DB, IR, ML, HCI), where each paper is associated with a bag of words in the abstract after removing the stop words.
- Cora Dataset[3]: A citation network consisting of 2708 scientific publications classified into one of seven classes.

The statistics of these two datasets is shown in Table 3. The original networks are directed and we symmetize the networks to make them undirected networks.

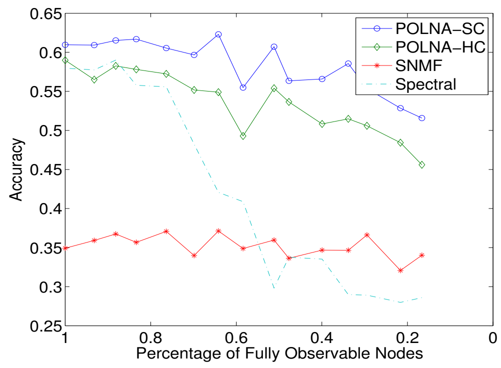
[3]http://linqs.cs.umd.edu/projects//projects/lbc/index.html



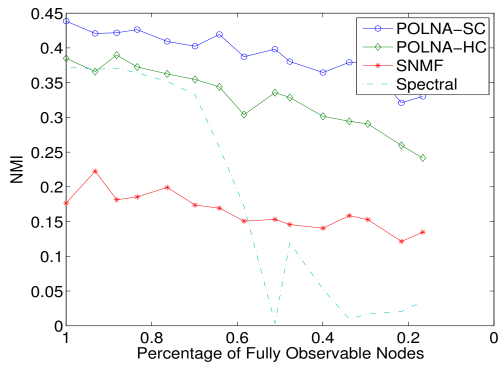
(a) Accuracy on Citeseer



(b) NMI on Citeseer



(c) Accuracy on Cora



(d) NMI on Cora

Figure 3: Community detection performance with different levels of fully observable nodes

B. *Experimental Setting*

1) *Baselines:* Existing methods usually assume the completeness of links and node attributes and thereby are not directly applicable. We create content links by constructing kNN graph for nodes with observable attributes. Then we construct a new network by connecting two nodes when they have either a structural link or content link between them. Two classic community detection methods, Symmetric Nonnegative Matrix Factorization (SNMF) [20] and Spectral Clustering [13], are applied to this combined network for community detection.

2) *Evaluation Metrics:* We use Accuracy and Normalized Mutual Information (NMI) to evaluate the quality of identified communities. Accuracy is measured as follows.

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n}\mathcal{I}(c_i = map(p_i)) \tag{20}$$

where p_i is the community detection result of data point i and c_i is its ground truth label. $map(\cdot)$ is a permutation mapping function that maps p_i to a class label using Kuhn-Munkres Algorithm [10].

Normalized Mutual Information (NMI) is calculated as follows. Let C be the set of communities from the ground truth and C' is obtained from a community detection algorithm.

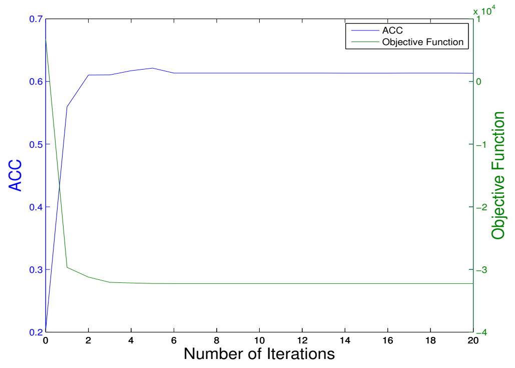
$$NMI(C, C') = \frac{MI(C, C')}{max(H(C), H(C'))} \tag{21}$$

where $H(C)$ and $H(C')$ are the entropy of C and C' and $MI(C, C')$ is the mutual information. Higher value of NMI indicates better quality of community detection.

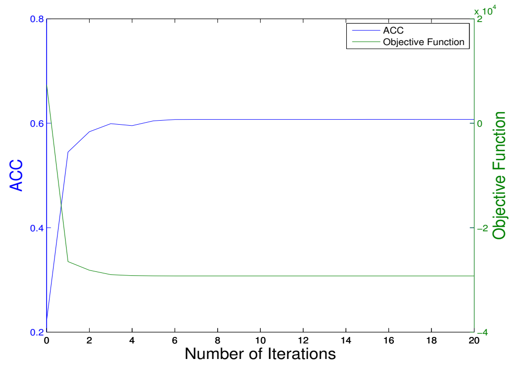
We use $k = 20$ for the kNN content network in our method and baselines. We set the number of latent factors in the proposed approach and baseline approaches as the number of ground-truth classes. For all methods, we perform kMeans[4] on the latent factors to obtain disjoint community affiliations. Since kMeans is affected by the initial seeds, we repeat the experiment for 20 times and report the average performance. For the proposed approach, we fix the regularization parameter $\lambda = 0.1$ and $\alpha = 1$ on both datasets and we study their sensitivity later in this paper.

3) *Generating Partially Observable Networks:* To create partially observable networks, we randomly select a few nodes (the number of these nodes is denoted as m_1) and remove their links. After removing these links, we denote

[4]We use the code at http://www.cad.zju.edu.cn/home/dengcai/Data/Clustering.html



(a) Convergence Analysis on Citeseer



(b) Convergence Analysis on Cora

Figure 4: Objective function and clustering performance at different numbers of iterations

the number of nodes without any links as m_2. Typically m_2 is larger than m_1 since removing the links for the m_1 nodes may also make some other nodes become isolated. Then we pick another m_2 nodes randomly which have links and remove their attributes. As a result, only $|O^s| = n - 2m_2$ nodes have both links and attributes.

C. Results

We report the community detection performance with different percentages (i.e., $|O^s|/n = 0.1 \sim 1.0$) of fully observable nodes in Figure 3. When comparing POLNA-SC with POLNA-HC, we observe that POLNA-SC usually outperforms POLNA-HC by $2\% \sim 10\%$. It indicates that rather than enforcing exactly the same value for link-based and attribute-based representations, it is preferable to allow some slight difference in $\mathbf{U}^g$ and $\mathbf{U}^a$. This is perhaps because links and attributes, though sharing significant common information, might also contain certain unique information. Compared with other baselines, the performance of both POLNA-SC and POLNA-HC decreases more gracefully when lowering the fully observable rate. For example, compared with using complete information, POLNA-SC with about $|O^s|/n = 50\%$ only decreases by 2.2% and 9.8% in accuracy on Cora and Citeseer datasets, respectively. When the fully observable rate goes to as low as 20%, POLNA-SC and POLNA-HC can still have decent community detection performance. In contrast, spectral clustering with combined structural and content links can achieve decent performance when the information is nearly complete, but its performance decreases significantly when fully observable rate is lower than 70%. This suggests the proposed partial co-regularization approach is more robust to incomplete information.

D. Convergence Analysis

Our optimization algorithm for POLNA-SC is based on alternating update on $\mathbf{U}^g$, $\mathbf{U}^a$ and $\mathbf{U}^*$. One might be concerned about how many outer iterations it needs to converge. We show the objective function and clustering performance (when about 50% of nodes are fully observable) in Figure 4. It can be observed that the optimization method converges very fast and can usually achieve stable performance in less than 10 iterations.

E. Sensitivity Analysis

In our algorithm, there are two regularization parameters α and λ for POLNA-SC and one regularization parameter λ for POLNA-HC. In this subsection, we investigate how the proposed methods perform with different parameter values.

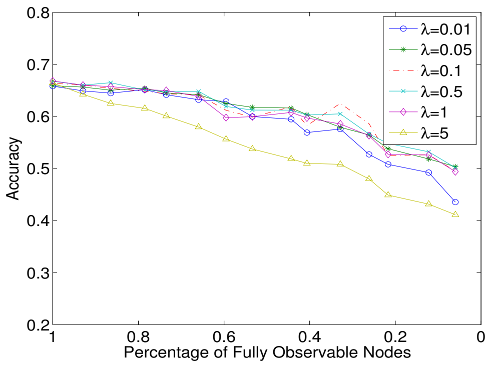
1) Parameters of POLNA-SC: For POLNA-SC, the accuracy results on two datasets with respect to different parameter values are shown in Figure 5.

- For λ, we can observe that POLNA is not very sensitive to the parameter value and performs consistently well when it is not too large (e.g., $\lambda \geq 5$) or too small (e.g., $\lambda \leq 0.05$).
- For the co-regularization parameter α, POLNA can achieve good performance as long as α is not too small (e.g., $\alpha \leq 0.1$). This is because when α is small, the co-regularization has so little effect that $\mathbf{U}^g$ and $\mathbf{U}^a$ can hardly benefit from the consensus learning. In other words, $\mathbf{U}^g$ lends insufficient strength to $\mathbf{U}^a$ (and vice versa), since the 'information bridge' (i.e., the co-regularization) is not given sufficient weight.

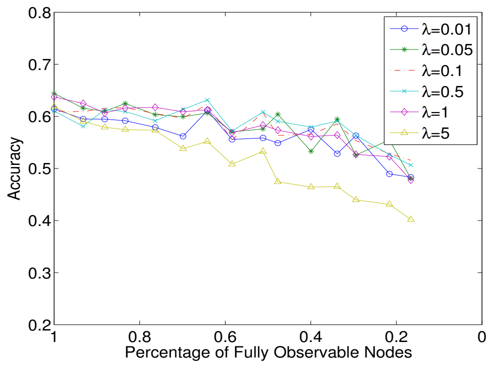
2) Parameters of POLNA-HC: The accuracy results for POLNA-HC with respect to different parameter values are shown in Figure 6. Similar to POLNA-SC, POLNA-HC is not very sensitive to λ and performs consistently well when λ is in a reasonable range.

VII. Conclusion

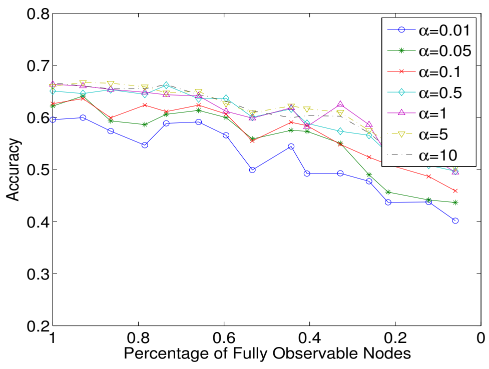
Many real-world social and information networks are only partially observable in terms of links and node attributes, which makes existing community detection methods not applicable or less effective. In this paper, we propose a



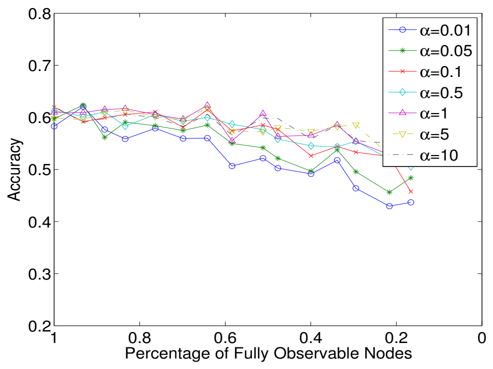
(a) Sensitivity of λ on Citeseer



(b) Sensitivity of λ on Cora



(c) Sensitivity of α on Citeseer



(d) Sensitivity of α on Cora

Figure 5: Parameter sensitivity for POLNA-SC

novel method, POLNA, for community detection on partially observable networks. The framework of POLNA is based on learning latent representations via kernel alignment technique, and a consensus of representation is learned on the nodes with both observable links and node attributes. Two variants of the framework are presented: POLNA-HC and POLNA-SC, which enforce hard and soft constraint on the consensus, respectively. Experimental results show that the proposed approach can perform reasonably well when there is even a small fraction of nodes having both observable links and node attributes. In future, we would like to explore detecting overlapping communities for partially observable networks.

Acknowledgement

This work is supported in part by NSF through grants IIS-1526499, and CNS-1626432. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X GPU used for this research.

References

[1] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. In *NIPS*, pages 33–40, 2009.

[2] R. Balasubramanyan and W. W. Cohen. Block-lda: Jointly modeling entity-annotated text and entity-entity links. In *SDM*, volume 11, pages 450–461. SIAM, 2011.

[3] R. Balasubramanyan and W. W. Cohen. Block-lda: Jointly modeling entity-annotated text and entity-entity links. In *SDM*, pages 450–461, 2011.

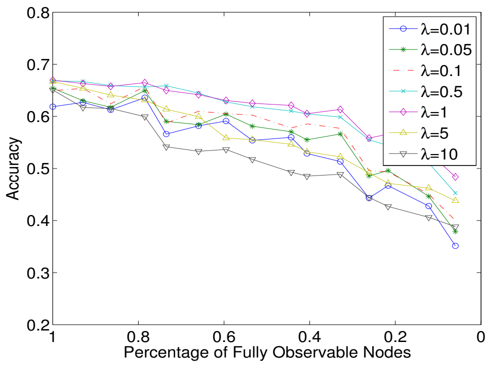
[4] D. Cai, X. He, J. Han, and T. S. Huang. Graph regularized non-negative matrix factorization for data representation. *PAMI*, 33(8):1548–1560, 2011.

[5] J. Chang and D. M. Blei. Relational topic models for document networks. In *AISTATS*, pages 81–88, 2009.

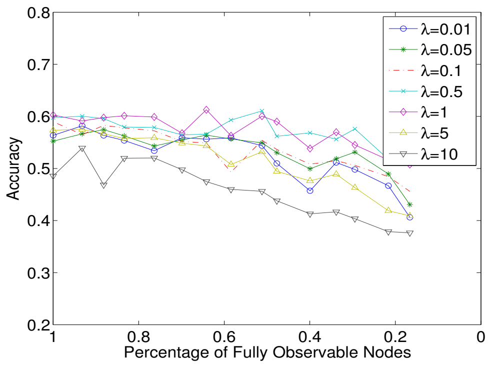
[6] M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi. Demon: a local-first discovery method for overlapping communities. In *KDD*, pages 615–623. ACM, 2012.

[7] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. S. Kandola. On kernel-target alignment. In *NIPS*, pages 367–373, 2001.

[8] A. Gretton, O. Bousquet, A. J. Smola, and B. Schlkopf. Measuring statistical dependence with hilbert-schmidt norms. In *ALT*, volume 3734 of *Lecture Notes in Computer Science*, pages 63–77. Springer, 2005.



(a) Sensitivity of λ on Citeseer



(b) Sensitivity of λ on Cora

Figure 6: Parameter sensitivity for POLNA-HC

[9] B. Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83(1):016107+, 2011.

[10] H. W. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, 1955.

[11] W. Lin, X. Kong, P. S. Yu, Q. Wu, Y. Jia, and C. Li. Community detection in incomplete information networks. In *WWW*, pages 341–350. ACM, 2012.

[12] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, Feb. 2004.

[13] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, pages 849–856, 2001.

[14] Y. Pei, N. Chakraborty, and K. P. Sycara. Nonnegative matrix tri-factorization with graph regularization for community detection in social networks. In *IJCAI*, pages 2083–2089, 2015.

[15] S. Sahebi and W. W. Cohen. Community-based recommendations: a solution to the cold start problem. In *Workshop on Recommender Systems and the Social Web*, 2011.

[16] P. Sen, G. M. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

[17] L. Song, A. Smola, A. Gretton, and K. Borgwardt. A dependence maximization view of clustering. In *ICML*, pages 815–822. ACM, 2007.

[18] A. P. Streich, M. Frank, D. Basin, and J. M. Buhmann. Multi-assignment clustering for boolean data. In *ICML*, pages 969–976. ACM, 2009.

[19] P. Tseng. Convergence of a Block Coordinate Descent Method for Nondifferentiable Minimization. *Journal of Optimization Theory and Applications*, 109(3), 2001.

[20] F. Wang, T. Li, X. Wang, S. Zhu, and C. Ding. Community discovery using nonnegative matrix factorization. *Data Min. Knowl. Discov.*, 22(3):493–521, 2011.

[21] X. Wei, B. Cao, and P. S. Yu. Nonlinear joint unsupervised feature selection. In *SDM*, 2016.

[22] X. Wei, B. Cao, and P. S. Yu. Unsupervised feature selection on networks: A generative view. In *AAAI*, 2016.

[23] X. Wei, S. Xie, and P. S. Yu. Efficient partial order preserving unsupervised feature selection on networks. In *SDM*, pages 82–90, 2015.

[24] Z. Xu, Y. Ke, Y. Wang, H. Cheng, and J. Cheng. A Model-based Approach to Attributed Graph Clustering. In *SIGMOD*, pages 505–516, 2012.

[25] J. Yang and J. Leskovec. Overlapping community detection at scale: a nonnegative matrix factorization approach. In *WSDM*, pages 587–596, 2013.

[26] J. Yang, J. J. McAuley, and J. Leskovec. Community detection in networks with node attributes. In *ICDM*, pages 1151–1156, 2013.

[27] T. Yang, R. Jin, Y. Chi, and S. Zhu. Combining link and content for community detection: a discriminative approach. In *KDD*, pages 927–936. ACM, 2009.

[28] Y. Zhou, H. Cheng, and J. X. Yu. Graph clustering based on structural/attribute similarities. *PVLDB*, 2:718–729, 2009.