A Bayesian Approach Toward Finding Communities and Their Evolutions in Dynamic Social Networks

Tianbao Yang[1] Yun Chi[2] Shenghuo Zhu[2] Yihong Gong[2] Rong Jin[1]
[1]Department of Computer Science and Engineering, Michigan State University, MI 48824, USA
[2]NEC Laboratories America, 10080 N. Wolfe Rd, SW3-350, Cupertino, CA 95014, USA
[1]{yangtia1,rongjin}@msu.edu, [2]{ychi,zsh,ygong}@sv.nec-labs.com

Abstract

Although a large body of work are devoted to finding communities in static social networks, only a few studies examined the dynamics of communities in evolving social networks. In this paper, we propose a dynamic stochastic block model for finding communities and their evolutions in a dynamic social network. The proposed model captures the evolution of communities by explicitly modeling the transition of community memberships for individual nodes in the network. Unlike many existing approaches for modeling social networks that estimate parameters by their most likely values (i.e., point estimation), in this study, we employ a Bayesian treatment for parameter estimation that computes the posterior distributions for all the unknown parameters. This Bayesian treatment allows us to capture the uncertainty in parameter values and therefore is more robust to data noise than point estimation. In addition, an efficient algorithm is developed for Bayesian inference to handle large sparse social networks. Extensive experimental studies based on both synthetic data and real-life data demonstrate that our model achieves higher accuracy and reveals more insights in the data than several state-of-the-art algorithms.

keywords: Social Network, Community, Community Evolution, Dynamic Stochastic Block Model, Bayesian Inference, Gibbs Sampling

1 Introduction

As online social networks such as Facebook and MySpace gaining popularity rapidly, social networks have become an ubiquitous part of many people's daily lives. Therefore, social network analysis is becoming a more and more important research field. One major topic in social network analysis is the study of communities in social networks. Analyzing communities in a social network, in addition to serving scientific purposes (e.g., in sociology and social psychology), helps improve user experiences (e.g., through friend recommendation services) and provides business values (e.g., in target advertisement and market segmentation analysis).

Communities have long been studied in various social networks. For example, in social science an important research topic is to identify cohesive subgroups of individuals within a social network where cohesive subgroups are defined as "subsets of actors among whom there are relatively strong, direct, intense, frequent, or positive ties" ([16]). As another example, communities also play an important role in Web analysis, where a Web community is defined as "a set of sites that have more links to members of the community than to non-members" ([7]).

Social networks are usually represented by graphs where nodes represent individuals and edges represent relationships and interactions among individuals. Based on this graph representation, there exists a large body of work on analyzing communities in *static* social networks, ranging from well-established social network analysis [16] to recent successful applications such as Web community discovery [7]. However, these studies overlooked an important feature of communities—communities in real life are usually dynamic. On a macroscopic level, community structures evolve over time. For example, a political community whose members' main interest is the presidential election may become less active after the election takes place. On a microscopic level, individuals may change their community memberships, due to the shifts of their interests or due to certain external events. In this respect, the above studies that analyze static communities fail to capture the important dynamics in communities.

Recently, there have been a growing body of work on analyzing *dynamic* communities in social networks. As we will discuss in detail in related work, some of these studies adopted a two-step approach where first static analysis is applied to the snapshots of the social network at different time steps, and then community evolutions are introduced afterwards to interpret the change of communities over time. Because data in real world are often noisy, such a two-step approach often results in unstable community structures and consequentially, unwarranted community evolutions. Some more recent studies attempted to unify the processes of community extraction and evolution extraction by using certain heuristics, such as regularizing *temporal smoothness*. Although some encouraging results were reported, none of these studies explicitly model the transition or change of community memberships, which is the key to the analysis of dynamic social network. In addition, most existing approaches consider point estimation in their studies, i.e., only estimate the most likely value for the unknown parameters. Given the large scale of social networks and potential noise in data, it is likely that the network data may not be sufficient to determine the exact value of parameters, and therefore it is important to develop methods beyond point estimation in order to model and capture the uncertainty in parameter estimation.

In this paper, we present a probabilistic framework for analyzing dynamic communities in social networks that explicitly addresses the above two problems. Instead of employing an afterwards effect or a regularization term, the proposed approach provides a unified framework for modeling both communities and their evolution simultaneously; the dynamics of communities is modeled explicitly by transition parameters that dictates the changes in community memberships over time; a Bayesian treatment of parameter estimation is employed to avoid the shortcoming of point estimation by using the posterior distributions of parameters for membership prediction. In short, we summarize the contributions of this work as follows.

- We propose a dynamic stochastic block model for modeling communities and their evolutions in a unified probabilistic framework. Our framework has two versions, the *online leaning* version that iteratively updates the probabilistic model over time, and the the *offline learning* version that learns the probabilistic model with network data obtained at all time steps. This is in contrast to most existing studies of social network analysis that only focus on the online learning approaches. We illustrate the advantage of the offline learning approach in our empirical study.

- We present a Bayesian treatment for parameter estimation in the proposed framework. Unlike most existing approaches for social network analysis that only computes the most likely values for the unknown parameters, the Bayesian treatment estimates the posterior distributions for unknown parameters, which is utilized to predict community memberships as well as to derive important characteristics of communities, such as community structures, community evolutions, etc.

- We develop a very efficient algorithm for the proposed framework. Our algorithm is executed in an incremental fashion to minimize the computational cost. In addition, our algorithm is designed to fully take advantage of the sparseness of data. We show that for each iteration, our algorithm has a time complexity linear in the size of a social network provided the network is sparse.

We conduct extensive experimental studies on both synthetic data and real data to investigate the performance of our framework. We show that compared to state-of-the-art baseline algorithms, our model is advantageous in (a) achieving better accuracy in community extraction, (b) capturing community evolutions more faithfully, and (c) revealing more insights from the network data.

2 Related Work

Finding communities is an important research topic in social network analysis. For the task of community discovery, many approaches such as clique-based, degree-based, and matrix-perturbation-based, have been proposed. Wasserman et al. [16] gave a comprehensive survey on these approaches. Community discovery is also related to some important research issues in other fields. For example, in applied physics, communities are important in analyzing modules in a physical system and various algorithms, such as [14], have been proposed to discover modular structures in physical systems. As another example, in the machine learning field, finding communities is closely related to graph-based clustering algorithms, such as the normalized cut algorithm proposed by Shi et al. [15] and the graph-factorization clustering (GFC) algorithm proposed by Yu et al. [17]. However, all these approaches focused on analyzing *static* networks while our focus in this study is on analyzing *dynamic* social networks.

In the field of statistics, a well-studied probabilistic model is the stochastic block model (SBM). This model had been originally proposed by Holland et al. [11] and have been successfully applied in various areas such as bioinformatics and social science [1, 6]. Researchers have extended the stochastic block model in different directions. For example, Airoldi et al. [1] proposed a mixed-membership stochastic block model, Kemp et al. [12] proposed a model that allows an unbounded number of clusters, and Hofman et al. [10] proposed a Bayesian approach based on the stochastic block model to infer module assignments and to identify the optimal number of modules. Our new model is also an extension of the stochastic block model. However, in comparison to the above approaches which focused on *static* social networks, our approach explicitly models the change of community membership over time and therefore can discovery communities and their evolutions simultaneously in dynamic social networks.

Recently, finding communities and their evolutions in dynamic networks has gained more and more attention. Asur et al. [2] introduced a family of events on both communities and individuals to characterize evolution of communities. Chi et al. [5] proposed an evolutionary version of the spectral clustering algorithms. They used graph cut as a metric for measuring community structures and community evolutions. Lin et al. [13] extended the graph-factorization clustering (GFC) and proposed the FacetNet algorithm for analyzing dynamic communities. We will conduct performance studies to compare our algorithm with some of these algorithms. Here we want to point out that compared to our new algorithm, none of these existing approaches has a rigorous probabilistic interpretation and they all are restricted to an online learning framework.

3 The Dynamic Stochastic Block Model

Before discussing the statistical models, we first introduce the notations that are used throughout this paper. We represent by $W^{(t)} \in \mathbb{R}^{n\times n}$ the *snapshot* of a social network at a given time step t (or snapshot network), where n is the number of nodes in the network. Each element w_{ij} in $W^{(t)}$ is the weight assigned to the link between nodes i and j: it can be the frequency of interactions (i.e., a natural number) or a binary number indicating the presence or absence of interactions between nodes i and j. For the time being, we focus on the binary link, which will be extended to other types of links in Section 6. For a dynamic social network, we use $\mathcal{W}_T = \{W^{(1)}, W^{(2)}, \ldots, W^{(T)}\}$ to denote a collection of snapshot graphs for a given social network over T discrete time steps. In our analysis and modeling, we first assume nodes in the social network remain unchanged during all the time steps, followed by the extension to dynamic social networks where nodes can be removed from and added to networks.

We use a $z_i \in \{1, \cdots, K\}$, where K is the total number of communities, to denote the community assignment of node i and we refer to z_i as the *community* of node i. We furthermore introduce $z_{ik} = [z_i = k]$ to indicate if node i is in the kth community where $[x]$ output one if x is true and zero otherwise. Community assignments matrix $Z = (z_{ik} : i \in \{1, \cdots, n\}, k \in \{1, \cdots, K\})$ includes the community assignments of all the nodes in a social network at a given time step. Finally, we use $\mathcal{Z}_T = \{Z^{(1)}, \ldots, Z^{(T)}\}$ to denote the collection of community assignments of all nodes over T time steps.

3.1 Dynamic Stochastic Block Model (DSBM)

We first briefly review the Stochastic Block Model (SBM). SBM is a well studied statistical model that has been successfully used in social network analysis [11]. In the SBM model, a network is generated in the following way. First, each node is assigned to a community following a probability $\pi = \{\pi_1, \ldots, \pi_K\}$ where π_k is the probability for a node to be assigned to community k. Then, depending on the community assignments of nodes i and j (assuming that $z_{ik} = 1$ and $z_{jl} = 1$), the link between i and j is generated following a Bernoulli distribution with parameter P_{kl}. So the parameters of SBM are $\pi \in \mathbb{R}^K$ and $P \in \mathbb{R}^{K\times K}$. The diagonal element P_{kk} of P is called the "within-community" link probability for community k and the off-diagonal element $P_{kl}, k \neq l$ is called "between-community" link probability between communities k and l.

Dynamic Stochastic Block Model (DSBM) extends SBM to dynamic social networks. It is defined in a recursive way. Assuming the community matrix $Z^{(t-1)}$ for time step *t-1* is available, we use a transition matrix $A \in \mathbb{R}^{K\times K}$ to model the community matrix $Z^{(t)}$ at time step t in the following way. For a node i, if $z_{ik}^{(t-1)} = 1$, i.e., node i was assigned to community k at time *t-1*, then with probability A_{kk} node i will remain in community k at time step t and with probability A_{kl} node i will change to another community l where $k \neq l$. We have each row of A sums to 1, i.e., $\sum_l A_{kl} = 1$. Given the community memberships in $Z^{(t)}$, the link between nodes will be then decided stochastically by probabilities in P as the SBM model. The generative process of the Dynamic Stochastic Block Model and the graphical representation are shown in Table 1 and Figure 1, respectively. Note that DSBM and SBM differ in how the community assignments are determined. In our DSBM model, instead of following a prior distribution π, the community assignments at any time t ($t > 1$) are determined by those at time *t-1* through transition matrix A, where A aims to capture the dynamic evolutions of communities.

Table 1: Generative Process of DSBM

```
For time 1:
  generate the Social Network followed by SBM
For each time $t > 1$:
  generate $z_i^{(t)} \sim p(z_i^{(t)} | z_i^{(t-1)}, A)$
For each pair $(i, j)$ at time $t$:
  generate $w_{ij}^{(t)} \sim Beronulli(\cdot | P_{z_i^{(t)}, z_j^{(t)}})$
```

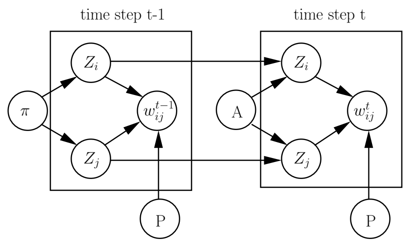


Figure 1: Graphical representation of Dynamic Stochastic Block Model (DSBM)

3.2 Likelihood of the Complete Data To express the data likelihood for the proposed DSBM model, we make two assumptions about the data generation process. First, link weight w_{ij} is generated independent of the other nodes/links provided membership z_i and z_j. Second, the community assignment $z_i^{(t)}$ of node i at time step t is independent of the other nodes/links provided its community assignment $z_i^{(t-1)}$ at time *t-1*. Using these assumptions, we write the likelihood of the complete data for our DSBM model as follows

$$\Pr(\mathcal{W}_T, \mathcal{Z}_T|\pi, P, A) =$$
$$\prod_{t=1}^{T} \Pr(W^{(t)}|Z^{(t)}, P) \prod_{t=2}^{T} \Pr(Z^{(t)}|Z^{(t-1)}, A) \Pr(Z^{(1)}|\pi)$$

where the emission probability $\Pr(W^{(t)}|Z^{(t)}, P)$ and the transition probability $\Pr(Z^{(t)}|Z^{(t-1)}, A)$ are

$$\Pr(W^{(t)}|Z^{(t)}, P) = \prod_{i\sim j} \Pr(w_{ij}^{(t)}|z_i^{(t)}, z_j^{(t)}, P)$$
$$= \prod_{i\sim j}\prod_{k,l} \left(P_{kl}^{w_{ij}^{(t)}}(1-P_{kl})^{1-w_{ij}^{(t)}}\right)^{z_{ik}^{(t)} z_{jl}^{(t)}}$$

and

$$\begin{aligned}\Pr(Z^{(t)}|Z^{(t-1)}, A) &= \prod_{i=1}^{n} \Pr(z_i^{(t)}|z_i^{(t-1)}, A)\\ &= \prod_{i=1}^{n}\prod_{k,l} A_{kl}^{z_{ik}^{(t-1)} z_{il}^{(t)}},\end{aligned}$$

respectively. Note that in our model, self-loops are not considered and so in the above equations, $i \sim j$ means over all i's and j's such that $i \neq j$. Finally, term $\Pr(Z^{(1)}|\pi)$ is the probability of community assignments at the first time step and is expressed as

$$\Pr(Z^{(1)}|\pi) = \prod_{i=1}^{n}\prod_{k} \pi_k^{z_{ik}^{(1)}}.$$

4 Bayesian Inference

In order to predict memberships of nodes in a given dynamic social network, a straightforward approach is to first estimate the most likely values for parameters π, P, and A from the historical data, and then infer the community memberships in the future using the estimated parameters. This is usually called point estimation in statistics, and is notorious for its instability when data is noisy. We address the limitation of point estimation by Bayesian inference [3]. Instead of using the most likely values for the model parameters, we utilize the distribution of model parameters when computing the prediction.

4.1 The Bayesian Model

Conjugate prior for Bayesian Inference We first introduce the prior distributions for model parameters π, P, and A. The conjugate prior for π is the Dirichlet distribution

$$\Pr(\pi) = \frac{\Gamma(\sum_k \gamma_k)}{\prod_k \Gamma(\gamma_k)} \prod_k \pi_k^{\gamma_k - 1} \tag{4.1}$$

where $\Gamma(\cdot)$ is the Gamma function. For the P matrix, we first assume it to be symmetric and therefore reduce the number of parameters to $\frac{n(n+1)}{2}$. The conjugate prior for each parameter P_{kl} for $l \geq k$ is a Beta distribution, and therefore the prior distribution for P is

$$\Pr(P) = \prod_{k,l\geq k} \frac{\Gamma(\alpha_{kl}+\beta_{kl})}{\Gamma(\alpha_{kl})\Gamma(\beta_{kl})} P_{kl}^{\alpha_{kl}-1}(1-P_{kl})^{\beta_{kl}-1}. \tag{4.2}$$

Finally, the conjugate prior for each row A is a Dirichlet distribution and the prior distribution for A is

$$\Pr(A) = \prod_k \frac{\Gamma(\sum_l \mu_{kl})}{\prod_l \Gamma(\mu_{kl})} \prod_l A_{kl}^{\mu_{kl}-1}. \tag{4.3}$$

Joint probability of the complete data To make our presentation concise, we introduce the following notations.

$$n_k^{(t)} = \sum_i z_{ik}^{(t)} \tag{4.4}$$

$$n_{k\to l}^{(t_1:t_2)} = \sum_{t=t_1+1}^{t_2} \sum_{i=1}^{n} z_{ik}^{(t-1)} z_{il}^{(t)} \tag{4.5}$$

$$n_{k\to \cdot}^{(t_1:t_2)} = \sum_{t=t_1+1}^{t_2} \sum_{i=1}^{n} z_{ik}^{(t-1)} \tag{4.6}$$

$$n_{kl}^{(t_1:t_2)} = \sum_{t=t_1}^{t_2} \sum_{i\sim j} (z_{ik}^{(t)} z_{jl}^{(t)} + z_{il}^{(t)} z_{jk}^{(t)}) \tag{4.7}$$

$$\hat{n}_{kl}^{(t_1:t_2)} = \sum_{t=t_1}^{t_2} \sum_{i\sim j} w_{ij}^{(t)} (z_{ik}^{(t)} z_{jl}^{(t)} + z_{il}^{(t)} z_{jk}^{(t)}) \tag{4.8}$$

Using these notations, and with the prior distributions of the model parameters, the theorem below gives the closed form expression for the joint probability of the complete data that is marginalized over the distribution of model parameters.

THEOREM 1. *With the priors of parameters $\theta = \{\pi, P, A\}$ defined in Equations (4.1)$\sim$(4.3) together with the notations given in Equations (4.4)$\sim$(4.8), the joint probability of observed links and unobserved community assignments is proportional to*

$$\Pr(\mathcal{W}_T, \mathcal{Z}_T) = \int \Pr(\mathcal{W}_T, \mathcal{Z}_T|\theta)\Pr(\theta)d\theta \propto$$

$$\prod_k \Gamma(n_k^{(1)} + \gamma_k) \prod_k \frac{\prod_l \Gamma(n_{k\to l}^{(1:T)} + \mu_{kl})}{\Gamma(n_{k\to\cdot}^{(1:T)} + \sum_l \mu_{kl})} \times$$
$$\prod_{k,l>k} B\left(\hat{n}_{kl}^{(1:T)} + \alpha_{kl}, n_{kl}^{(1:T)} - \hat{n}_{kl}^{(1:T)} + \beta_{kl}\right) \times$$
$$\prod_k B\left(\frac{\hat{n}_{kk}^{(1:T)}}{2} + \alpha_{kk}, \frac{n_{kk}^{(1:T)} - \hat{n}_{kk}^{(1:T)}}{2} + \beta_{kk}\right)$$

where $B(\cdot)$ is the Beta function.

Due to the limit of space, we skip the detailed proof. In this Bayesian inference framework, to obtain the community assignment of each node at each time step, we need to compute the posterior probability $\Pr(\mathcal{Z}_T|\mathcal{W}_T)$. This is in general an intractable problem. In the next two subsections, we introduce two versions of the inference method, i.e., an offline learning approach and an online learning approach.

4.2 Offline learning In offline learning, it is assumed that the link data of all time steps are accessible and therefore, the community assignments of all nodes in all time steps can be decided simultaneously by maximizing the posterior probability, i.e.,

$$\mathcal{Z}_T^* = \arg\max_{\mathcal{Z}_T} \Pr(\mathcal{Z}_T|\mathcal{W}_T) = \arg\max_{\mathcal{Z}_T} \Pr(\mathcal{W}_T, \mathcal{Z}_T) \tag{4.9}$$

where $\Pr(\mathcal{W}_T, \mathcal{Z}_T)$ is given in Theorem 1. Note that in offline learning, the community membership of each node at every time step t is decided by the link data of all time steps, even the link data of time steps later than t. Given this observation, we expect offline learning to deliver more reliable estimation of community memberships than the online learning that is discussed in the next subsection.

4.3 Online learning In online learning, the community memberships are learned incrementally over time. Assume we have decided the community membership $Z^{(t-1)}$ at time step *t-1*, and observed the links $W^{(t)}$ at time t. We decide the community assignments at time t by maximizing the posterior probability of community assignments at time t given $Z^{(t-1)}$ and $W^{(t)}$, i.e.,

$$Z^{*(t)} = \arg\max_{Z^{(t)}} \Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)})$$

Hence, to decide $Z^{(t)}$, the key is to efficiently compute $\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)})$ except for time step 1 in which we need to compute $\Pr(Z^{(1)}|W^{(1)})$. The following theorem provides closed form solutions for the two probabilities. It is important to note that both probabilities are computed by averaging over the distribution of the model parameters.

THEOREM 2. *With the priors of parameters $\theta = \{\pi, P, A\}$ given in Equations (4.1)$\sim$(4.3), the posterior probability of unobserved community assignments given the observed links and the community assignments at previous time step is proportional to*

$$\Pr(Z_1|W_1) \propto \prod_k \Gamma(n_k^{(1)} + \gamma_k)$$
$$\times \prod_{k,l>k} B\left(\hat{n}_{kl}^{(1)} + \alpha_{kl}, n_{kl}^{(1)} - \hat{n}_{kl}^{(1)} + \beta_{kl}\right)$$
$$\times \prod_k B\left(\frac{\hat{n}_{kk}^{(1)}}{2} + \alpha_{kk}, \frac{n_{kk}^{(1)} - \hat{n}_{kk}^{(1)}}{2} + \beta_{kk}\right)$$

$$\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)}) \propto \tag{4.10}$$

$$\prod_k \left(\prod_l \frac{\Gamma(n_{k\to l}^{(t-1:t)} + \mu_{kl})}{\Gamma(n_{k\to\cdot}^{(t-1:t)} + \sum_l \mu_{kl})}\right)$$
$$\times \prod_{k,l>k} B\left(\hat{n}_{kl}^{(t)} + \alpha_{kl}, n_{kl}^{(t)} - \hat{n}_{kl}^{(t)} + \beta_{kl}\right)$$
$$\times \prod_k B\left(\frac{\hat{n}_{kk}^{(t)}}{2} + \alpha_{kk}, \frac{n_{kk}^{(t)} - \hat{n}_{kk}^{(t)}}{2} + \beta_{kk}\right).$$

We skip the detailed proof due to the limit of space. In online learning, it is assumed that data arrives sequentially and historic community assignments are not updated upon the arrival of new data. Therefore, the online learning algorithm can be implemented more efficiently than the offline learning algorithm.

5 Inference Algorithm

To optimize the posterior probabilities in the offline and online learning algorithms introduced in the previous section, we appeal to Gibbs sampling method. In Gibbs sampling, we need to compute the conditional probability of the community assignment of each node conditioned on the community assignments of other nodes. We will first describe the algorithm and then provide time complexity of the proposed algorithm.

5.1 Gibbs sampling algorithm For offline learning, we need to compute the conditional probability $\Pr(z_i^{(t)}|\mathcal{Z}_{T,\{i,t\}^-}, \mathcal{W}_T)$, via $\Pr(\mathcal{Z}_T|\mathcal{W}_T)$, where $\mathcal{Z}_{T,\{i,t\}^-}$ are the community assignments of all nodes at all time steps except node i at time step t. This can be computed by marginalizing $z_i^{(t)}$ in Equation (4.9). Similarly, for online learning, we need to compute the conditional probability $\Pr(z_i^{(t)}|Z_{i^-}^{(t)}, W^{(t)}, Z^{(t-1)})$, where $Z_{i^-}^{(t)}$ is the collection of community assignments of all nodes, except node i, at time step t. This can be computed by marginalizing $\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)})$. The following algorithms describe a simulated annealing version of our inference algorithm.

Algorithm 5.1. Probabilistic Simulated Annealing Algorithm

1. Randomly initialize the community assignment for each node at time step t (online learning) or at all time steps (offline learning); select the temperature sequence $\{T_1, \cdots, T_M\}$ and the iteration number sequence $\{N_1, \cdots, N_M\}$.
2. for each iteration $m = 1, \ldots, M$, run N_m iterations of Gibbs sampling with target distributions $\exp\{\log \Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)})/T_m\}$ or $\exp\{\log \Pr(\mathcal{Z}_T|\mathcal{W}_T)/T_m\}$.

Algorithm 5.2. Gibbs Sampling Algorithm

1. Compute the following statistics with the initial assignments:

$$n_k^{(1)}$$

$$n_{kl}^{(1:T)}, \hat{n}_{kl}^{(1:T)} \quad \textit{or} \quad n_{kl}^{(t)}, \hat{n}_{kl}^{(t)}$$

$$n_{k\to l}^{(1:T)}, n_{k\to\cdot}^{(1:T)} \quad \textit{or} \quad n_{k\to l}^{(t-1:t)}, n_{k\to\cdot}^{(t-1:t)}$$

2. for each iteration $m_i = 1 : N_m$, and for each node $i = 1 : n$ at each time t
   - Compute the objective function in Simulated Annealing

     $$\exp\left\{\log \Pr(z_i^t|Z_{i^-}^{(t)}, W^{(t)}, Z^{(t-1)})/T_m\right\} \quad \textit{or}$$

     $$\exp\left\{\log \Pr(z_i^t|\mathcal{Z}_{T,\{i,t\}^-}, \mathcal{W}_T)/T_m\right\}$$

     up to a constant using the current statistics, and then obtain the normalized distribution.
   - Sample the community assignment for node i according to the distribution obtained above, update it to the new one.
   - Update the statistics.

5.2 Time complexity In our implementation, we adopt several techniques to improve the efficiency of the algorithm. First, since in each step of the sampling, only one node i at a given time t changes its community assignment, almost all the statistics can be updated incrementally to avoid recomputing. Second, our algorithm is designed to take advantage of the sparseness of the matrix $W^{(t)}$. For instance, we exploit the sparseness of $W^{(t)}$ to facilitate the computation of $\hat{n}_{kl}^{(t_1:t_2)}$. Without details, we give the time complexity as the following.

Theorem 3. *The time complexity of our implementation of the Gibbs sampling algorithm is $O(nT + eT + K^2T + NT(eC_1 + nC_2))$ where e is the total number of edges in the social network over all the time steps, N is the number of iterations in Gibbs sampling , C_1 and C_2 are constants.*

As can be seen, when the social network is sparse and when the degree of each node is bounded by a constant, the running time of each iteration of our Gibbs sampling algorithm is linear in the size of the social network.

6 Extensions

In this section, we present two extensions to our basic framework, including how to handle different types of links and how to handle insertion and deletion of nodes in the network. In addition, we discuss how to choose the hyperparameters in our model.

6.1 Handling different types of link So far, we have used binary links in our model, where the binary links (i.e., either $w_{ij} = 1$ or $w_{ij} = 0$) indicate the presence or absence of a relation between a pair of nodes. However, there exist other types of links in social networks as well. Here we show how to extend our model to handle two other cases: when $w_{ij} \in \mathcal{N}$ and when $w_{ij} \in \mathcal{R}^+$. If w_{ij} indicates the frequency of interactions (e.g., the occurrence of interactions between two bloggers during a day, the number of papers that two authors co-authored during a year, etc.), then w_{ij} can be any non-negative integer. Our current model actually can handle this case with little change: the emission probability

$$\Pr(w_{ij}|z_i, z_j) = \prod_{k,l} \left(P_{kl}^{w_{ij}}(1 - P_{kl})\right)^{z_{ik}z_{jl}} \tag{6.11}$$

remains valid for $w_{ij} \in \mathcal{N}$, except that instead of a Bernoulli distribution (i.e., $w_{ij} = 0$ or 1), now w_{ij} follows a geometric distribution. Note that the $(1 - P_{kl})$ term is needed to take into account the case where there is no edge between i and j.

In other applications, w_{ij} represents the similarity or distance between nodes i and j and therefore $w_{ij} \in$

$\mathcal{R}^+$, the set of non-negative real numbers. In such a case, we can first discretize the w_{ij} by using finite bins and then introduce the emission probabilities as before. Another way to handle the case when $w_{ij} \in \mathcal{R}^+$ is suggested by Zhu [18], which is to introduce a k-nearest neighbor graph and therefore reduce the problem to the case when $w_{ij} = 0$ or 1.

6.2 Handling the variability of nodes In dynamic social networks, at a given time, new individuals may join in the network and old ones may leave. To handle insertion of new nodes and deletion of old ones, existing algorithm such as [5] and [13] use some heuristics, e.g., by assuming that all the nodes are in the network all the time but in some time steps certain nodes have no incident links. In comparison, in both the online and the offline versions of our algorithm, no such heuristics are necessary. For example, for online learning, let S_t denote the set of nodes at time t, $I_t = S_t \bigcap S_{t-1}$ be set of nodes appearing in both time steps t and $t-1$. $U_t = S_t - S_{t-1}$ be the new nodes at time t. Then we can naturally model the posterior probability of the community assignments at time t as

$$\text{(6.12)} \quad \Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)}) \propto \Pr(Z^{(t)}, W^{(t)}|Z^{(t-1)})$$
$$= \Pr(W^{(t)}|Z^{(t)}) \Pr(Z^{(t)}_{I_t}|Z^{(t-1)}_{I_t}) \Pr(Z^{(t)}_{U_t})$$

and we can directly write the part corresponding to Equation (4.10) in Theorem 2 as

$$\Pr(Z^{(t)}|W^{(t)}, Z^{(t-1)}) \propto$$
$$\prod_k \Gamma(n^{(t)}_{k,U_t} + \gamma_k) \times \prod_k \left(\prod_l \frac{\Gamma(n^{(t-1:t)}_{k\to l,I_t} + \mu_{kl})}{\Gamma(n^{(t-1:t)}_{k\to\cdot,I_t} + \sum_l \mu_{kl})} \right)$$
$$\times \prod_{k,l>k} B\left(\hat{n}^{(t)}_{kl,S_t} + \alpha_{kl}, n^{(t)}_{kl,S_t} - \hat{n}^{(t)}_{kl,S_t} + \beta_{kl}\right)$$
$$\times \prod_k B\left(\frac{\hat{n}^{(t)}_{kk,S_t}}{2} + \alpha_{kk}, \frac{n^{(t)}_{kk,S_t} - \hat{n}^{(t)}_{kk,S_t}}{2} + \beta_{kk}\right)$$

where $n^*_{*,S}$ is the corresponding statistics evaluated on the nodes set of S. Similar results can be derived for the offline learning algorithm. In brief, our model can handle the insertion and deletion of nodes without using any heuristics.

6.3 Hyperparameters In this section, we discuss the roles of the hyperparameters (γ, α, β, and μ) and give some guidelines on how to choose the values for these hyperparameters. In the experimental studies, we will further investigate the impact of the values of these hyperparameters on the performance of our algorithm.

γ is the hyperparameter for the prior of π. We can interpret the γ_k as an effective number of observations of $z_{ik} = 1$. Without other prior knowledge we set all γ_k to be the same. α, β are the hyperparameters for the prior of P. As stated before, we discriminate two probabilities in P, i.e., P_{kk} the "within-community" link probability, and $P_{kl,l\neq k}$ the "between-community" link probability. For the hyperparameters, we set two groups of values, i.e., (1) $\alpha_{kk}, \beta_{kk}, \forall k$ and (2) $\alpha_{kl,l\neq k}, \beta_{kl,l\neq k}$. Because we have the prior knowledge that nodes in the same community have higher probability to link to each other than nodes in different communities, we set $\alpha_{kk} \geq \alpha_{kl,l\neq k}, \beta_{kk} \leq \beta_{kl,l\neq k}$. μ is the hyperparameter for A. $A_{k*} = \{A_{k1}, \cdots, A_{kk}, \cdots, A_{kK}\}$ are the transition probabilities for nodes to switch from the kth community to other (including coming back to the kth) communities in the following time step. $\mu_{k*} = \{\mu_{k1}, \cdots, \mu_{kk}, \cdots, \mu_{kK}\}$ can be interpreted as effective number of nodes in the kth community switching to other (including coming back to the kth) communities in the following time step. With prior belief that most nodes will not change their community memberships over time, we set $\mu_{kk} \geq \mu_{kl,l\neq k}$.

Finally, how to select the exact values for the hyperparameters γ, α, β, and μ is described in the empirical studies.

7 Experimental Studies

In this section, we conduct several experimental studies. First, we show that the performance of our algorithms is not sensitive to most hyperparameters in the Bayesian inference and for the only hyperparameters that impact the performance significantly, we provide a principled method for automatic parameter selection. Second, we show that our Gibbs-sampling-based algorithms have very fast convergence rate, which makes our algorithms very practical for real applications. Third, by using a set of benchmark datasets with a variety of characteristics, we demonstrate that our algorithms clearly outperform several state-of-the-art algorithms in terms of discovering the true community memberships and capturing the true evolutions of community memberships. Finally, we use two real datasets of dynamic social networks to illustrate that from these datasets, our algorithms are able to reveal interesting insights that are not directly obtainable from other algorithms.

7.1 Performance Metrics The experiments we conducted can be categorized into two types, those with ground truth available and those without ground truth. By ground truth we mean the true community membership of each node at each time step. When the ground truth is available, we measure the performance of an algorithm by the *normalized mutual information* between the true community memberships and those given by the algorithm. More specifically, if the true community memberships are represented by $\mathcal{C} = \{C_1, \ldots, C_K\}$ and those given by the algorithm are represented by $\mathcal{C}' = \{C'_1, \ldots, C'_K\}$, then the *mutual information* between the two is defined as

$$\widehat{MI}(\mathcal{C}, \mathcal{C}') = \sum_{C_i, C'_j} p(C_i, C'_j) \log \frac{p(C_i, C'_j)}{p(C_i)p(C'_j)}$$

and the *normalized mutual information* is defined by

$$MI(\mathcal{C}, \mathcal{C}') = \frac{\widehat{MI}(\mathcal{C}, \mathcal{C}')}{\max(H(\mathcal{C}), H(\mathcal{C}'))}$$

where $H(\mathcal{C})$ and $H(\mathcal{C}')$ are the entropies of the partitions $\mathcal{C}$ and $\mathcal{C}'$. The value of MI is between 0 and 1 and a higher MI value indicates that the result given by the algorithm $\mathcal{C}'$ is closer to the ground truth $\mathcal{C}$. This metric MI has been commonly used in the information retrieval field [9].

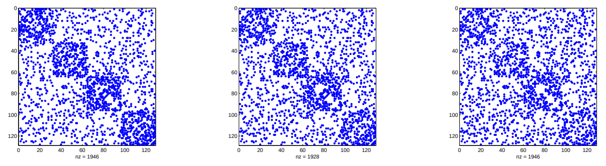
Where there is no ground truth available in the dataset, we measure the performance by using the metric of *modularity* which is proposed by Newman et al. [14] for measuring community partitions. For a given community partition $\mathcal{C} = \{C_1, \ldots, C_K\}$, the modularity is defined as

$$Modu(\mathcal{C}) = \sum_k \left[\frac{Cut(V_k, V_k)}{Cut(V, V)} - \left(\frac{Cut(V_k, V)}{Cut(V, V)} \right)^2 \right]$$

where V represents all the nodes in the social network and V_k indicates the set of nodes in the kth community C_k. $Cut(V_i, V_j) = \sum_{p \in V_i, q \in V_j} w_{pq}$. As state in [14], modularity measures how likely a network is generated due to the proposed community structure versus generated by a random process. Therefore, a higher modularity value indicates a community structure that better explains the observed social network. Many existing studies, such as [4, 13], have used this metric for performance analysis.

7.2 Experiments on Synthetic Datasets

7.2.1 Data generator We generate the synthetic data by following a procedure suggested by Newman et al. [14]. The data consists of 128 nodes that belong to 4 communities with 32 nodes in each community. Links are generated in the following way. For each pair of nodes that belong to the same community, the probability that a link exists between them is p_{in}; the probability that a link exists between a pair of nodes belonging to different communities is p_{out}. However, by fixing the average degree of nodes in the network, which we set to be 16 in our datasets, only one of p_{in} and p_{out} can change freely. By increasing p_{out}, the network becomes more noisy in the sense that the community structure becomes less obvious and hard to detect. We generate datasets under three different noise levels by setting p_{in}=0.1452 (p_{out}=0.0365), p_{in}=0.1290 (p_{out}=0.0417), and p_{in}=0.1129 (p_{out}=0.0469), respectively. The ratio of p_{out}/p_{in} increases from 0.2512 for level one to 0.3229 for level two and 0.4152 for level three. The adjacency matrices for the datasets of the three noise levels are shown in Figure 2.



(a) Level 1 (b) Level 2 (c) Level 3

Figure 2: The adjacency matrices for the datasets with different noise levels.

The above network generator described by Newman et al. can only generate static networks. To study dynamic evolution, we let the community structure of the network evolve in the following way. We start with introducing evolutions to the community memberships: at each time step after time step 1, we randomly choose 10% of the nodes to leave their original community and join the other three communities at random. After the community memberships are decided, links are generated by following the probabilities p_{in} and p_{out} as before. We generate the network with community evolution in this way for 10 time steps.

7.2.2 Hyperparameters In the first experiment, we study the impact of the hyperparameters on the performance of our algorithm. Figure 3 shows the performance of our algorithm, in terms of the average normalized mutual information and the average modularity over all time steps, under a large range of values for the hyperparameters γ (for the initial probability π) and μ (for the transition matrix A), respectively. As can be seen, the performance varies little under different values for γ and μ, which verifies that our algorithm is robust to the setting of these hyperparameters. As a result, in the following experiments, unless stated otherwise, we simply set $\gamma = 1$ and $\mu_{kk} = 10$. Note that we only show the results of our online learning algorithm for the dataset with noise level 2. The results for the dataset with other noise levels and for the offline learning algorithm are similar and therefore are not shown here.

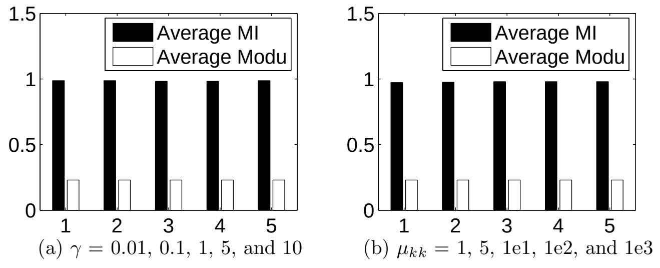


Figure 3: The performance, in terms of the average normalized mutual information and the average modularity over all time steps, under different values for γ and μ_{kk} (with $\mu_{kl} = 1, \forall k \neq l$), which shows that the performance is not sensitive to γ and μ.

However, the performance of our algorithm is somewhat sensitive to the hyperparameters α and β for P, which is the stochastic matrix representing the community structure at each time step. In Figure 4 we show the performance of our algorithm under a large range of α and β values, which demonstrates that the performance varies under different α and β values. This result makes sense because α and β are crucial for the stochastic model to correctly capture the community structure of the network. For example, the best performance is achieved when α is in the same range as the total number of links in the network. In addition, we see a clear correlation between the accuracy with respect to the ground truth, which is not seen by our algorithm, and the modularity, which is available to our algorithm. As a result, we can use the modularity value as a validation metric to automatically choose good values for α and β. All the experimental results reported in the following are obtained from this automatic validation procedure.

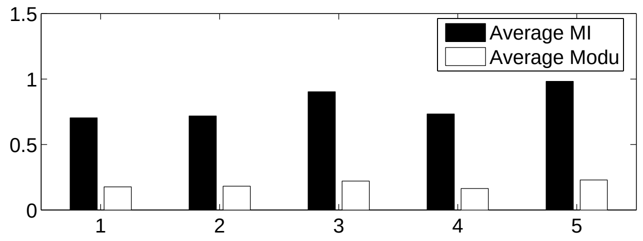


Figure 4: The performance, in terms of the average normalized mutual information and the average modularity over all time steps, under the cases with α and β valued at α, β are ($\alpha_{kk} = 1, \beta_{kl} = 1$), ($\alpha_{kk} = 5, \beta_{kl} = 1$), ($\alpha_{kk} = 10, \beta_{kl} = 1$), ($\alpha_{kk} = 1e2, \beta_{kl} = 10$), ($\alpha_{kk} = 1e4, \beta_{kl} = 10$), and in all the cases $\alpha_{kl,l\neq k}$=1.

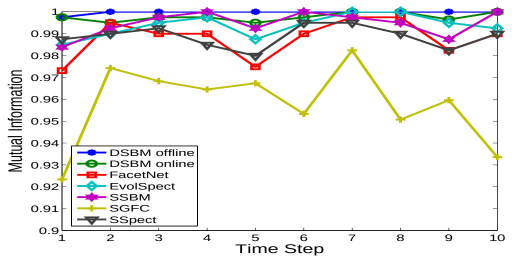
7.2.3 Comparison with the baseline algorithms

In this experiment, we compare the performance of the online and offline versions of our DSBM algorithm with those of two recently proposed algorithms for analyzing dynamic communities—the dynamic graph-factorization clustering algorithm (FacetNet) by Lin et al. [13] and the evolutionary spectral clustering algorithm (EvolSpect) by Chi et al. [5]. In addition, we also provide the performances of the static versions for all the algorithms—static stochastic block models (SSBM) for DSBM, static graph-factorization clustering (SGFC) for FacetNet, and static spectral clustering (SSpect) for EvolSpect.

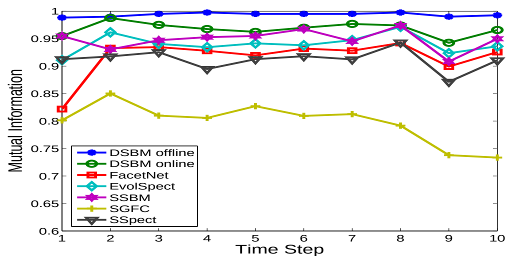
Figure 5 presents the performance, in terms of the normalized mutual information with respect to the ground truth over the 10 time steps, of all the algorithms for the three datasets with different noise levels. We can obtain the following observations from the results. First, our DSBM algorithms have the best accuracy and outperform all other baseline algorithms at every time step for all the three datasets. Second, the offline version of our algorithm, which takes into consideration all the available data simultaneously, has better performance than that of the online version. Third, the evolutionary versions of all the algorithms outperform their static counterparts in most cases, which demonstrates the advantages of the dynamic models in capturing community evolutions in dynamic social networks. We obtain similar results for the modularity metric and due to the limit of space, we do not include them here.

Next, we investigate which algorithms can capture the community evolution more faithfully. For this purpose, since we have the ground truth on which nodes actually changed their communities at each time step, we compute the precision (the fraction of nodes, among those an algorithm found switched communities, that really changed their communities) and the recall (the fraction of nodes, among those really changed their communities, that an algorithm found switched communities) for all the algorithms. Figure 6 shows the average precision and recall for all the algorithms over all the time steps for the three datasets with different noise levels. As can be seen, our DSBM algorithms have the best precision and the best recall values for all the three datasets, which illustrates that our algorithms can capture the true community evolution more faithfully than the baseline algorithms.

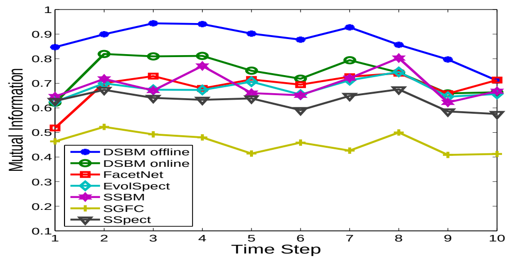
7.2.4 Convergence rate In our algorithms, we adopt the Gibbs sampling for Bayesian inference. In this experiment, we show that this Gibbs sampling procedure converges very quickly. In Figure 7, we show how the value of the objective function changes over the number of iterations at each time step. As can be seen,



(a) noise of level 1



(b) noise of level 2



(c) noise of level 3

Figure 5: The normalized mutual information with respect to the ground truth over the 10 time steps, of all the algorithms on the three datasets with different noise levels.

the first time step requires more iterations but even for the first time step, fewer than 20 iterations are enough for the algorithm to converge. For the time steps 2 to 10, by using the results at the previous time step as the initial values, the algorithm converges in just a couple of iterations. This result, together with the time complexity analysis in Section 5.2, demonstrates that our algorithm is practical and is scalable to large social networks in real applications.

7.3 Experiments on Real Datasets

In this section we present experimental studies on two real datasets: a traditional social network dataset, and a blog dataset.

7.3.1 The southern women data

The southern women data is a standard benchmark data in social science. It was collected in 1930's in Natchez, Mississippi.

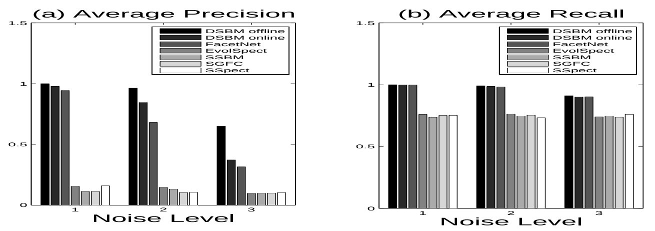


Figure 6: (a) The average precision and (b) the average recall over all the time steps for the three datasets with different noise levels.

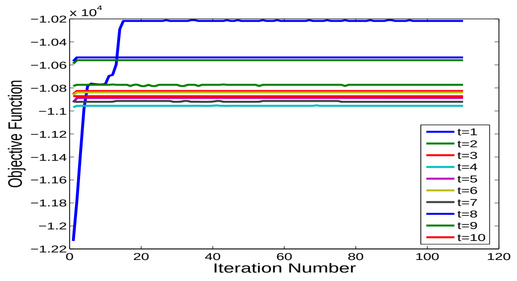
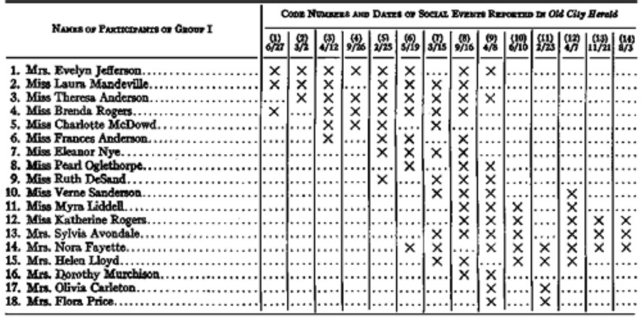


Figure 7: Convergence rate of Gibbs sampling procedure in the online learning.

The data records the attendance of 18 women in 14 social events during a period of one year. The detailed data is presented in Table 2. We obtain the social network by assigning w_{ij} for women i and j the number of times that they co-participated in the same events. We first apply the static stochastic model (SBM) to the aggregated data and we set the number of communities to be 2, the number used in most previous studies. Not surprisingly, we obtain the same result as most social science methods reported in [8], that is, women 1–9 belong to one community and women 10–18 belong to the other community.

Next, based on the number of events that occurred, we partition the time period into 3 time steps: (1)

Table 2: The southern women data, from [8].



Names of Participants of Group I	Code Numbers and Dates of Social Events Reported in *Old City Herald*													
	(1) 6/27	(2) 3/2	(3) 4/12	(4) 9/26	(5) 2/25	(6) 5/19	(7) 3/15	(8) 9/16	(9) 4/8	(10) 6/10	(11) 2/23	(12) 4/7	(13) 11/21	(14) 8/3
1. Mrs. Evelyn Jefferson	×	×	×	×	×	×		×	×					
2. Miss Laura Mandeville	×	×	×		×	×	×	×						
3. Miss Theresa Anderson		×	×	×	×	×	×	×	×					
4. Miss Brenda Rogers	×		×	×	×	×	×	×						
5. Miss Charlotte McDowd			×	×	×		×							
6. Miss Frances Anderson			×		×	×		×						
7. Miss Eleanor Nye					×	×	×	×						
8. Miss Pearl Oglethorpe						×		×	×					
9. Miss Ruth DeSand					×		×	×	×					
10. Miss Verne Sanderson							×	×	×			×		
11. Miss Myra Liddell								×	×	×		×		
12. Miss Katherine Rogers								×	×	×		×	×	×
13. Mrs. Sylvia Avondale							×	×	×	×		×	×	×
14. Mrs. Nora Fayette						×	×		×	×	×	×	×	×
15. Mrs. Helen Lloyd							×	×		×	×	×		
16. Mrs. Dorothy Murchison								×	×					
17. Mrs. Olivia Carleton									×		×			
18. Mrs. Flora Price									×		×			

February–March, when women 1–7,9,10, and 13–18, participated social events 2,5,7, and 11; (2) April–May, when women 8,11,12, and 16 joined in and together they participated in events 3,6,9, and 12; (3) June–November, when events 1,4,8,10, and 13 happened for which women 17 and 18 did not show up. We apply both the offline and the online versions of our algorithm on this dataset with 3 time steps. It turns out that the offline algorithm reports no community change for any woman. This result suggests that if we take the overall data into consideration simultaneously, the evidence is not strong enough to justify any change of community membership. However, in the online learning algorithm, if we decrease the hyperparameter μ_{kk} for A to a very small value (around 1) and therefore encourage changes of community memberships, women 6–9 start to change their community at time step 3. By investigating the data in Table 2 we see that this change is due to the social event 8, which is the only event that women 6–9 participated at time step 3 and is mainly participated by women who were not in the same community as women 6–9 at time steps 1 and 2.

7.3.2 The Blog Dataset This blog dataset was collected by the NEC Labs and have been used in several previous studies on dynamic social networks [5, 13]. It contains 148,681 entry-to-entry links among 407 blogs during 15 months. In this study, we first partition the data in the following way. The first 7 months are used for the first 7 time steps; data in months 8 and 9 are aggregated into the 8th time step; data in months 10–15 are aggregated into the 9th time step. The reason for this partition is that in the original dataset, the number of links dropped dramatically toward the end of the time and the partition above makes the number of links at each time step to be evenly around 200.

Clustering performance We run our algorithms on this dataset and compare the performance, in terms of modularity, with that of the two baselines, the dynamic graph-factorization clustering (FacetNet [13]) and the evolutionary spectral clustering (EvolSpect [5]). Following [5], we set the number of communities to be 2 (which roughly correspond to a technology community and a political community). In terms of hyperparameters for our algorithm, for γ and μ, we simply chose some default values (i.e., $\gamma_k = 1$, $\mu_{kl} = 1$, and $\mu_{kk} = 10$), and for α and β, we chose the ones that result in the best modularity. For the two baseline algorithms, their parameters are chosen to obtain the best modularity. Figure 8 shows the performance of the algorithms. As can be seen, for this dataset, the offline and online versions of our algorithm give similar results and they both outperform the baseline algorithms.

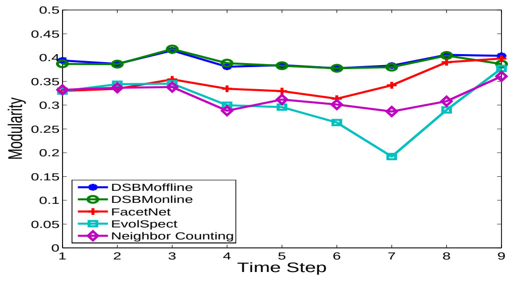


Figure 8: The performance, in terms of the modularity, of different algorithms (including the naive method using neighbor counting) on the NEC blog datasets.

Some meaningful community changes Actually, we found that most blogs are stable in terms of their communities. However, there are still some blogs changing their communities detected by our algorithms based on the links information. Here, we present the community memberships of four representative blogs. Three of them (blogs 94, 192, and 357) have the most number of links across the whole time and one of them (blog 230) has the least number of links, only at two time steps. To help the visualization, we assign one of the two labels to each blog where the labels are obtained by applying the normalized cut algorithm [15] on the aggregated blog graph. Therefore, these labels give us the community membership of each blog if we use *static* analysis on the *aggregated data*. Then to visualize the *dynamic* community memberships, for a blog at a given time step, we show the fractions of the blog's neighbors (through links) that have each of the two possible labels at the given time step.

Figure 9 illustrates how these fractions change over time for these 4 representative blogs. On the top of each subfigure, we show the community memberships computed by our algorithms and at the bottom of each subfigure, we show some top keywords that occurred most frequently in the corresponding blog site. From the figure we can see that blogs 94 and 132, for which our algorithms detect no changes in community memberships, have very homogeneous neighbor labels and very focused top keywords. In comparison, blog 357 has relatively inhomogeneous neighbors during different time steps and has more changes of community memberships detected by our algorithm. It turns out that blog 357 is a blog that reports news events in the area of San Francisco, including both technology events and political events. Finally, we look at blog 230. This blog is more technology focused. However, during the whole time period, it only generated 2 entry-to-entry links in time steps 3 and 8 respectively. The link generated at time 3 pointed to a political blog (blog 60, `http://catallarchy.net`) and that at time 8 pointed to a technology blog (blog 362, `http://www.siliconbeat.com`). Although the results obtained by our algorithms are correct based on these evidence, we can see that the link-based analysis can be unreliable when the links are very sparse.

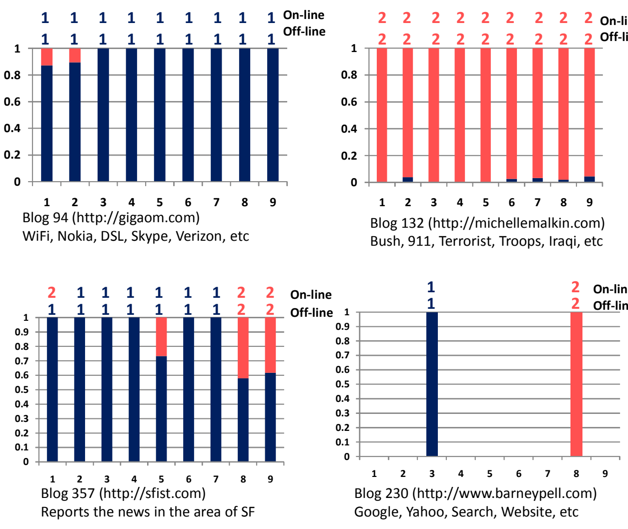


Figure 9: Neighbor distributions of the four representative blogs, the community results of the offline and online versions of DSBM, and some top keywords occurred in the blogs.

Now seeing that the simple neighbor counting approach gives results consistent with our algorithms, one may wonder if such a naive approach is good enough for analyzing dynamic social networks. To answer this question, in Figure 8 we also plot the performance of this neighbor counting approach and we can see that such a naive approach by itself does not give performances comparable to our algorithms.

8 Conclusion

In this paper, we proposed a framework based on Bayesian inference to find communities and to capture community evolutions in dynamic social networks. The framework is a probabilistic generative model that unifies the communities and their evolutions in an intuitive and rigorous way; the Bayesian treatment gives robust prediction of community memberships; the proposed algorithms are implemented efficiently to make them practical in real applications. Extensive experimental studies showed that our algorithms outperform several state-of-the-art baseline algorithms in different measures and reveal very useful insights in several real social networks.

References

[1] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic block models for relational data with application to protein-protein interactions. In *Proc. of the International Biometrics Society Annual Meeting*, 2006.

[2] S. Asur, S. Parthasarathy, and D. Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. In *Proc. of the 13th ACM SIGKDD Conference*, 2007.

[3] C. M. Bishop. *Pattern recognition and machine learning (information science and statistics).* Springer-Verlag New York, Inc., 2006.

[4] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner. On modularity clustering. *IEEE Trans. on Knowl. and Data Eng.*, 20(2), 2008.

[5] Y. Chi, X. Song, D. Zhou, K. Hino, and B. L. Tseng. Evolutionary spectral clustering by incorporating temporal smoothness. In *Proc. of the 13th ACM SIGKDD Conference*, 2007.

[6] S. E. Fienberg, M. M. Meyer, and S. S. Wasserman. Statistical analysis of multiple sociometric relations. *J. of the American Statistical Association*, 80(389), 1985.

[7] G. Flake, S. Lawrence, and C. Giles. Efficient identification of web communities. In *Proc. of the 6th ACM SIGKDD Conference*, 2000.

[8] L. C. Freeman. *Finding social groups: A meta-analysis of the southern women data.* The National Academies Press, 2003.

[9] Y. Gong and W. Xu. *Machine Learning for Multimedia Content Analysis.* Springer, 2007.

[10] J. M. Hofman and C. H. Wiggins. A Bayesian approach to network modularity. *Phy. Rev. Letters*, 100, 2008.

[11] P. Holland and S. Leinhardt. Local structure in social networks. *Socialogical Methodology*, 1976.

[12] C. Kemp, T. L. Griffiths, and J. B. Tenenbaum. Discovering latent classes in relational data. Technical Report AI Memo 2004-019, MIT Computer Science and Artificial Intelligence Laboratory, 2004.

[13] Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng. FacetNet: a framework for analyzing communities and their evolutions in dynamic networks. In *Proc. of the 17th WWW Conference*, 2008.

[14] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phy. Rev. E*, 69(2), 2003.

[15] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8), 2000.

[16] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications.* Cambridge University Press, 1994.

[17] K. Yu, S. Yu, and V. Tresp. Soft clustering on graphs. In *NIPS*, 2005.

[18] X. Zhu. *Semi-supervised learning with graphs.* PhD thesis, Carnegie Mellon University, 2005.