Generalized Latent Factor Models for Social Network Analysis

Wu-Jun Li[†], **Dit-Yan Yeung**[‡], **Zhihua Zhang**[♯]
[†] Shanghai Key Laboratory of Scalable Computing and Systems
Department of Computer Science and Engineering, Shanghai Jiao Tong University, China
[‡] Department of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong, China
[♯] College of Computer Science and Technology, Zhejiang University, China
liwujun@cs.sjtu.edu.cn, dyyeung@cse.ust.hk, zhzhang@cs.zju.edu.cn

Abstract

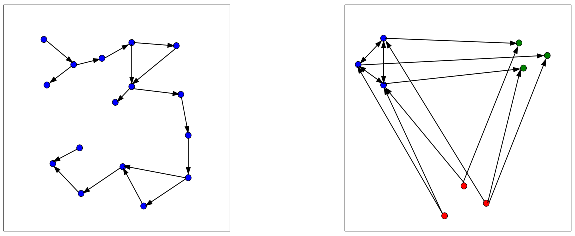
Homophily and *stochastic equivalence* are two primary features of interest in social networks. Recently, the *multiplicative latent factor model* (MLFM) is proposed to model social networks with *directed* links. Although MLFM can capture stochastic equivalence, it cannot model well homophily in networks. However, many real-world networks exhibit homophily or both homophily and stochastic equivalence, and hence the network structure of these networks cannot be modeled well by MLFM. In this paper, we propose a novel model, called *generalized latent factor model* (GLFM), for social network analysis by enhancing homophily modeling in MLFM. We devise a *minorization-maximization* (MM) algorithm with linear-time complexity and convergence guarantee to learn the model parameters. Extensive experiments on some real-world networks show that GLFM can effectively model homophily to dramatically outperform state-of-the-art methods.

1 Introduction

A social network [1] [Wasserman and Faust, 1994] is often represented as a graph in which the nodes represent the objects and the edges (or called links) represent the binary relations between objects. The edges in a graph can be directed or undirected. If the edges are directed, we call the graph a directed graph. Otherwise, the graph is an undirected graph. Unless otherwise stated, we focus on directed graphs in this paper because an undirected edge can be represented by two directed edges with opposite directions. Some typical networks include friendship networks among people, web graphs, and paper citation networks.

As pointed out by [Hoff, 2008], *homophily* and *stochastic equivalence* are two primary features of interest in social networks. If an edge is more likely to exist between two nodes with similar characteristics than between those nodes having different characteristics, we say the graph exhibits homophily. For example, two individuals are more likely to be friends if they share common interests. Hence, a friendship graph has the feature of homophily. On the other hand, if the nodes of a graph can be divided into groups where members within a group have similar patterns of links, we say this graph exhibits stochastic equivalence. The web graph has such a feature because some nodes can be described as hubs which are connected to many other nodes called authorities but the hubs or authorities are seldom connected among themselves. For stochastic equivalence, the property that members within a group have similar patterns of links also implies that if two nodes link to or are linked by one common node, the two nodes most likely belong to the same group.

Examples of homophily and stochastic equivalence in directed graphs are illustrated in Figure 1, where the locations in the 2-dimensional space denote the characteristics of the points (nodes). From Figure 1(a), we can see that a link is more likely to exist between two points close to each other, which is the property of homophily. In Figure 1(b), the points form three groups associated with different colors, and the nodes in each group share similar patterns of links to nodes in other groups, but the nodes in the same group are not necessarily connected to each other. This is the property of stochastic equivalence. Note that in a graph exhibiting stochastic equivalence, two points close to each other are not necessarily connected to each other and connected points are not necessarily close to each other, which is different from the property of homophily.



(a) homophily (b) stochastic equivalence

Figure 1: Homophily and stochastic equivalence in networks.

[1] In this paper, we use the terms 'network', 'social network' and 'graph' interchangeably.

As social network analysis (SNA) is becoming more and more important in a wide range of applications, many SNA models have been proposed [Goldenberg *et al.*, 2009]. In this paper, we focus on *latent variable models* [Bartholomew and Knott, 1999] which have been successfully applied to model social networks [Hoff, 2008; Nowicki and Snijders, 2001; Hoff *et al.*, 2002; Kemp *et al.*, 2006; Airoldi *et al.*, 2008; Hoff, 2009]. These models include: the latent class model [Nowicki and Snijders, 2001] and its extensions [Kemp *et al.*, 2006; Airoldi *et al.*, 2008], the latent distance model [Hoff *et al.*, 2002], the latent eigenmodel [Hoff, 2008], and the multiplicative latent factor model (MLFM) [Hoff, 2009]. Among all these models, the recently proposed latent eigenmodel, which includes both the latent class model and the latent distance model as special cases, can capture both homophily and stochastic equivalence in networks. However, it can only model *undirected* graphs. MLFM [Hoff, 2009] adapts the latent eigenmodel for *directed* graphs. However, as to be shown in our experiments, in fact it cannot model well homophily.

In this paper, we propose a novel model, called *generalized latent factor model* (GLFM), for social network analysis by enhancing homophily modeling in MLFM. The learning algorithm of GLFM is guaranteed to converge to a local optimum and has linear-time complexity. Hence, GLFM can be used to model large-scale graphs. Extensive experiments on community detection in some real-world networks show that GLFM dramatically outperforms existing methods.

2 Notation and Definition

We use boldface uppercase letters, such as $\mathbf{K}$, to denote matrices, and boldface lowercase letters, such as $\mathbf{x}$, to denote vectors. The ith row and the jth column of a matrix $\mathbf{K}$ are denoted as $\mathbf{K}_{i*}$ and $\mathbf{K}_{*j}$, respectively. K_{ij} denotes the element at the ith row and jth column in $\mathbf{K}$. x_i denotes the ith element in $\mathbf{x}$. We use $\mathrm{tr}(\mathbf{K})$ to denote its trace, $\mathbf{K}^T$ for its transpose and $\mathbf{K}^{-1}$ for its inverse. $\|\cdot\|$ is used to denote the length of a vector. $|\cdot|$ denotes the cardinality of a set. $\mathbf{I}$ denotes the identity matrix whose dimensionality depends on the context. For a matrix $\mathbf{K}$, $\mathbf{K} \succeq 0$ means that $\mathbf{K}$ is positive semi-definite (psd) and $\mathbf{K} \succ 0$ means that $\mathbf{K}$ is positive definite (pd). $\mathbf{K} \succeq \mathbf{M}$ means $\mathbf{K} - \mathbf{M} \succeq 0$. $\mathcal{N}(\cdot)$ denotes the normal distribution, either for scalars or vectors. $\circ$ denotes the Hadamard product (element-wise product).

Let N denote the number of nodes in a graph. $\mathbf{A}$ is the adjacency (link) matrix for the N nodes. $A_{ij} = 1$ if there exists a link from node i to node j. Otherwise, $A_{ij} = 0$. D denotes the number of latent factors. In real-world networks, if $A_{ij} = 1$, we can say that there is a relation from i to j. However, $A_{ij} = 0$ does not necessarily mean that there is no relation from i to j. In most cases, $A_{ij} = 0$ means that the relationship from i to j is missing. Hence, we use an indicator matrix $\mathbf{Z}$ to indicate whether or not an element is missing. More specifically, $Z_{ij} = 1$ means that A_{ij} is observed while $Z_{ij} = 0$ means that A_{ij} is missing.

3 Multiplicative Latent Factor Models

The latent eigenmodel is formulated as follows [2]:

$$\Theta_{ik} = \log \mathrm{odds}(A_{ik} = 1 \mid \mathbf{X}_{i*}, \mathbf{X}_{k*}, \mu) = \mu + \mathbf{X}_{i*}\mathbf{\Lambda}\mathbf{X}_{k*}^T,$$

where $\mathbf{X}$ is an $N \times D$ matrix with $\mathbf{X}_{i*}$ denoting the latent representation of node i and μ is a parameter reflecting the overall density of the links in the network, $\mathbf{\Lambda}$ is a $D \times D$ diagonal matrix with the diagonal entries being either positive or negative. The latent eigenmodel generalizes both latent class models and latent distance models. It can model both homophily and stochastic equivalence in *undirected* graphs [Hoff, 2008].

To adapt the latent eigenmodel for *directed* graphs, MLFM defines

$$\Theta_{ik} = \mu + \mathbf{X}_{i*}\mathbf{\Lambda}\mathbf{W}_{k*}^T, \tag{1}$$

where $\mathbf{X}$ and $\mathbf{W}$ have orthonormal columns. Note that the key difference between the latent eigenmodel and MLFM lies in the fact that MLFM adopts a different *receiver* factor matrix $\mathbf{W}$ which enables MLFM to model directed (asymmetric) graphs. As we will show in our experiments, this modification in MLFM makes it fail to model homophily in networks.

Letting $\mathbf{\Theta} = [\Theta_{ik}]_{i,k=1}^N$, we can rewrite MLFM as follows:

$$\mathbf{\Theta} = \mu\mathbf{E} + \mathbf{X}\mathbf{\Lambda}\mathbf{W}^T, \tag{2}$$

where $\mathbf{E}$ is an $N \times N$ matrix with all entries being 1.

We can find that MLFM is a special case of the following model:

$$\mathbf{\Theta} = \mu\mathbf{E} + \mathbf{U}\mathbf{V}^T. \tag{3}$$

For example, we can get MLFM by setting $\mathbf{U} = \mathbf{X}$ and $\mathbf{V} = \mathbf{W}\mathbf{\Lambda}$. Furthermore, it is easy to compute the $\mathbf{X}$, $\mathbf{W}$ and $\mathbf{\Lambda}$ in (2) based on the learned $\mathbf{U}$ and $\mathbf{V}$ in (3). Hence, in the sequel, MLFM refers to the model in (3).

4 Generalized Latent Factor Models

As discussed above, MLFM can capture stochastic equivalence but cannot model well homophily in directed graphs. Here, we propose our GLFM to enhance homophily modeling in MLFM.

4.1 Model

In GLFM, Θ_{ik} is defined as follows:

$$\Theta_{ik} = \mu + \frac{1}{2}\mathbf{U}_{i*}\mathbf{U}_{k*}^T + \frac{1}{2}\mathbf{U}_{i*}\mathbf{V}_{k*}^T. \tag{4}$$

Comparing (4) to (3), we can find that GLFM generalizes MLFM by adding an extra term $\mathbf{U}_{i*}\mathbf{U}_{k*}^T$.[3] It is

[2] Note that in this paper, we assume for simplicity that there is no attribute information for the links. It is straightforward to integrate attribute information into the existing latent variable models as well as our proposed model.

[3] Note that the coefficient $\frac{1}{2}$ in (4) makes no essential difference between (4) and (3). It is only for convenience of computation.

this extra term that enables GLFM to model homophily in networks, which will be detailed in Section 4.2 when we analyze the objective function in (7). This will also be demonstrated empirically later in our experiments.

Based on (4), the likelihood of the observations can be defined as follows:

$$p(\mathbf{A} \mid \mathbf{U}, \mathbf{V}, \mu) = \prod_{i \neq k} [S_{ik}^{A_{ik}} (1 - S_{ik})^{1-A_{ik}}]^{Z_{ik}}, \qquad (5)$$

where

$$S_{ik} = \frac{\exp(\Theta_{ik})}{1 + \exp(\Theta_{ik})}. \qquad (6)$$

Note that as in the conventional SNA model, we ignore the diagonal elements of $\mathbf{A}$. That is, in this paper, we set $A_{ii} = Z_{ii} = 0$ by default.

Furthermore, we put normal priors on the parameters μ, $\mathbf{U}$ and $\mathbf{V}$: $p(\mu) = \mathcal{N}(\mu \mid 0, \tau^{-1}), p(\mathbf{U}) = \prod_{d=1}^{D} \mathcal{N}(\mathbf{U}_{*d} \mid \mathbf{0}, \beta\mathbf{I}), p(\mathbf{V}) = \prod_{d=1}^{D} \mathcal{N}(\mathbf{V}_{*d} \mid \mathbf{0}, \gamma\mathbf{I})$.

4.2 Learning

Although the Markov chain Monte Carlo (MCMC) algorithms designed for other latent variable models can easily be adapted for GLFM, we do not adopt MCMC here for GLFM because MCMC methods typically incur very high computational cost. In this paper, we adopt the *maximum a posteriori* (MAP) estimation strategy to learn the parameters. The log posterior probability can be computed as follows:

$$\begin{aligned} L = & \sum_{i \neq k} \Big\{ \frac{1}{2} A_{ik} \mathbf{U}_{i*} \mathbf{U}_{k*}^T + \frac{1}{2} A_{ik} \mathbf{U}_{i*} \mathbf{V}_{k*}^T + A_{ik}\mu \\ & \qquad - Z_{ik} \log \left[1 + \exp(\Theta_{ik})\right] \Big\} \\ & - \frac{1}{2\beta} \mathrm{tr}(\mathbf{U}\mathbf{U}^T) - \frac{1}{2\gamma} \mathrm{tr}(\mathbf{V}\mathbf{V}^T) - \frac{\tau}{2}\mu^2 + c, \qquad (7) \end{aligned}$$

where c is a constant independent of the parameters. Note that in (7) we assume that all existing links should be observed. That is to say, if $A_{ik} = 1$, then $Z_{ik} = 1$.

The term $A_{ik}\mathbf{U}_{i*}\mathbf{U}_{k*}^T$ in (7) results from the extra term $\mathbf{U}_{i*}\mathbf{U}_{k*}^T$ in (4). In (7), to maximize the objective function L, we have to make $\mathbf{U}_{i*}\mathbf{U}_{k*}^T$ as large as possible if there exists a link between nodes i and k (i.e., $A_{ik} = 1$). This conforms to the property of homophily, i.e., a link is more likely to exist between two nodes with similar characteristics than between those nodes having different characteristics. Note that here the latent factor $\mathbf{U}_{i*}$ reflects the characteristics of node i. Therefore, the extra term $\mathbf{U}_{i*}\mathbf{U}_{k*}^T$ in (4) enables GLFM to model homophily in networks.

If we directly optimize all the parameters $\mathbf{U}$, $\mathbf{V}$ and μ jointly, the computational cost will be very high. For example, if we want to use the second-order information, generally we need to invert the Hessian matrix where the time complexity is cubic in the number of parameters.

Here, we adopt an alternating projection strategy to maximize L. More specifically, each time we optimize one parameter, such as $\mathbf{U}$, with the other parameters fixed.

Learning U

To learn $\mathbf{U}$, we optimize each row of it with all other rows fixed. The gradient vector and Hessian matrix can be computed as follows:

$$\begin{aligned} \frac{\partial L}{\partial \mathbf{U}_{i*}^T} = & - \frac{1}{\beta} \mathbf{U}_{i*}^T + \frac{1}{2} \mathbf{V}^T \Big[\mathbf{A}_{i*}^T - (\mathbf{Z}_{i*} \circ \mathbf{S}_{i*})^T \Big] \\ & + \frac{1}{2} \mathbf{U}^T \Big[\mathbf{A}_{i*}^T + \mathbf{A}_{*i} - (\mathbf{Z}_{i*} \circ \mathbf{S}_{i*})^T - \mathbf{Z}_{*i} \circ \mathbf{S}_{*i} \Big], \end{aligned}$$

$$\begin{aligned} \frac{\partial^2 L}{\partial \mathbf{U}_{i*}^T \partial \mathbf{U}_{i*}} = & - \frac{1}{\beta} \mathbf{I} - \frac{1}{4} \sum_{k, k \neq i} \Big\{ Z_{ki} S_{ki} (1 - S_{ki}) \mathbf{U}_{k*}^T \mathbf{U}_{k*} \Big\} \\ & - \frac{1}{4} \sum_{k, k \neq i} \Big\{ Z_{ik} S_{ik} (1 - S_{ik}) [\mathbf{U}_{k*} + \mathbf{V}_{k*}]^T [\mathbf{U}_{k*} + \mathbf{V}_{k*}] \Big\}. \end{aligned}$$

Because both the gradient vector and Hessian matrix depend on $\mathbf{S}_{i*}$ which is a function of $\mathbf{U}_{i*}$, we have to resort to iterative methods to find the optimal values. Here, we devise a *minorization-maximization* (MM) algorithm [Lang *et al.*, 2000] to learn it. MM is a so-called expectation-maximization (EM) algorithm [Dempster *et al.*, 1977] without missing data, alternating between constructing a concave lower bound of the objective function and maximizing that bound.

Because $0 < S_{ik} < \frac{1}{2}$, we can get $S_{ik}(1 - S_{ik}) < \frac{1}{4}$.

Let us define:

$$\begin{aligned} \mathbf{H}_i = & - \frac{1}{\beta} \mathbf{I} - \frac{1}{16} \sum_{k, k \neq i} \Big\{ Z_{ik} [\mathbf{U}_{k*} + \mathbf{V}_{k*}]^T [\mathbf{U}_{k*} + \mathbf{V}_{k*}] \Big\} \\ & - \frac{1}{16} \sum_{k, k \neq i} \Big\{ Z_{ki} \mathbf{U}_{k*}^T \mathbf{U}_{k*} \Big\}. \end{aligned}$$

It is easy to prove that $\frac{\partial^2 L}{\partial \mathbf{U}_{i*}^T \partial \mathbf{U}_{i*}} \succeq \mathbf{H}_i$.

Let

$$\begin{aligned} f(\mathbf{U}_{i*}) = & L(\mathbf{U}_{i*}(t)) + [\mathbf{U}_{i*} - \mathbf{U}_{i*}(t)] \times \frac{\partial L}{\partial \mathbf{U}_{i*}^T}(t) \\ & + \frac{1}{2} [\mathbf{U}_{i*} - \mathbf{U}_{i*}(t)] \mathbf{H}_i(t) [\mathbf{U}_{i*} - \mathbf{U}_{i*}(t)]^T, \end{aligned}$$

where $\mathbf{U}_{i*}(t)$ denotes the value of the former iteration and $\mathbf{H}_i(t)$ is computed with the updated $\mathbf{U}$ except for $\mathbf{U}_{i*}$.

Then we have the following theorem:

Theorem 1 $L(\mathbf{U}_{i*}) \geq f(\mathbf{U}_{i*})$, *which means that* $f(\mathbf{U}_{i*})$ *is a lower bound of* $L(\mathbf{U}_{i*})$.

The proof of Theorem 1 is simple and we omit it here.

We can see that $f(\mathbf{U}_{i*})$ has a quadratic form of $\mathbf{U}_{i*}$. By setting the gradient of $f(\mathbf{U}_{i*})$ with respect to $\mathbf{U}_{i*}$ to 0, we have the update rule for $\mathbf{U}_{i*}$:

$$\mathbf{U}_{i*}(t+1) = \mathbf{U}_{i*}(t) - \Big[\frac{\partial L}{\partial \mathbf{U}_{i*}^T}(t) \Big]^T \times \mathbf{H}_i(t)^{-1}.$$

Learning $\mathbf{V}$

The gradient vector and Hessian matrix of $\mathbf{V}_{i*}$ can be computed as follows:

$$\frac{\partial L}{\partial \mathbf{V}_{i*}^T} = -\frac{1}{\gamma}\mathbf{V}_{i*}^T + \frac{1}{2}\mathbf{U}^T\Big[\mathbf{A}_{*i} - (\mathbf{Z}_{*i} \circ \mathbf{S}_{*i})\Big]$$

$$\frac{\partial^2 L}{\partial \mathbf{V}_{i*}^T \partial \mathbf{V}_{i*}} = -\frac{1}{\gamma}\mathbf{I} - \frac{1}{4}\sum_{k,k\neq i}\Big\{Z_{ki}S_{ki}(1-S_{ki})\mathbf{U}_{k*}^T\mathbf{U}_{k*}\Big\}.$$

Let $\mathbf{G}_i = -\frac{1}{\gamma}\mathbf{I} - \frac{1}{16}\sum_{k,k\neq i}\Big\{Z_{ki}\mathbf{U}_{k*}^T\mathbf{U}_{k*}\Big\}$, we can prove that $\frac{\partial^2 L}{\partial \mathbf{V}_{i*}^T \partial \mathbf{V}_{i*}} \succeq \mathbf{G}_i$.

Similar to the update rule for $\mathbf{U}_{i*}$, we can obtain the update rule for $\mathbf{V}_{i*}$ as follows:

$$\mathbf{V}_{i*}(t+1) = \mathbf{V}_{i*}(t) - \Big[\frac{\partial L}{\partial \mathbf{V}_{i*}^T}(t)\Big]^T \times \mathbf{G}_i(t)^{-1},$$

where $\mathbf{V}_{i*}(t)$ denotes the value of the former iteration and $\mathbf{G}_i(t)$ is computed with the updated parameters except for $\mathbf{V}_{i*}$.

Learning μ

Using similar learning techniques as those for $\mathbf{U}$ and $\mathbf{V}$, we can get the update rule for μ:

$$\mu(t+1) = \mu(t) + \frac{4[\sum_{k\neq i}(A_{ik} - Z_{ik}S_{ik}) - \tau\mu(t)]}{4\tau + \sum_{k\neq i} Z_{ik}}.$$

4.3 Convergence and Computational Complexity

With the MM algorithm, the learning procedure of GLFM is guaranteed to converge to a local maximum.

The time complexity to compute the gradient and Hessian for node i is linear to the total number of ones in both $\mathbf{Z}_{*i}$ and $\mathbf{Z}_{i*}$. In general, this number is $O(1)$ because the observations in real networks are always very sparse. Furthermore, since $\mathbf{H}_i$ and $\mathbf{G}_i$ are $D\times D$, the computational complexity to invert the Hessian matrices is $O(D^3)$. Typically, D is a very small number. Hence, to update the whole $\mathbf{U}$ and $\mathbf{V}$, only $O(N)$ time is needed.

5 Experiment

There exist many different SNA tasks such as social position and role estimation [Wasserman and Faust, 1994], link prediction [Hoff, 2009], node classification [Li *et al.*, 2009a; Li and Yeung, 2009; Li *et al.*, 2009b], community detection [Yang *et al.*, 2009a], and so on. In this paper, we adopt the same evaluation strategy as that in [Yang *et al.*, 2009a; 2009b] for social community detection. The main reason for choosing this task is that from our model formulation we can clearly see the difference between GLFM and other latent factor models. However, many other models from different research communities have also been proposed for SNA. It is difficult to figure out the connection and difference between those models and GLFM from the formulation perspective. Hence, we use empirical evaluation to compare them. Most mainstream models have been compared in [Yang *et al.*, 2009a; 2009b] for community detection, which provides a good platform for our empirical comparison.

For MLFM and GLFM, we adopt k-means to perform clustering based on the normalized latent representation $\mathbf{U}$. Here normalization means that the latent representation of each node is divided by its length. Because the magnitude of $\mathbf{U}_{i*}$ reflects the activity of i, we select the most active user as the first seed of the k-means, and then choose a point as the seed of the next community if summation of the distances between this point and all the existing seeds is the largest one. Hence, the initialization of k-means is fixed. We set $\mathbf{Z} = \mathbf{A}$. For fair comparison, the hyper-parameters in GLFM and all the baselines to be compared, such as the τ in (7), are chosen from a wide range and the best results are reported. More specifically, for GLFM, the τ is fixed to 10^6, β and γ are set to 2, and $D = 20$.

5.1 Data Sets and Evaluation Metric

As in [Yang *et al.*, 2009a], we use two paper citation networks, Cora and Citeseer data sets [4], for evaluation. Both data sets contain content information in addition to the directed links.

The Cora data set contains 2708 research papers from the 7 subfields of machine learning: case-based reasoning, genetic algorithms, neural networks, probabilistic methods, reinforcement learning, rule learning, and theory. Each paper is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from a dictionary of 1433 unique words. There are overall 5429 citations (links) between the papers.

The Citeseer data set contains 3312 papers which can be classified into 6 categories. Each paper is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from a dictionary of 3703 unique words. There are overall 4732 citations (links) between the papers. After deleting the self-links, we obtain 4715 links for our evaluation.

As in [Yang *et al.*, 2009a], we use *Normalized Mutual Information* (*NMI*), *Pairwise F-Measure* (*PWF*) and *Modularity* (*Modu*) as metrics to measure the clustering accuracy of our model. For all the algorithms, we set the number of communities to the ground-truth number of class labels in the data.

5.2 Baselines

We compare GLFM with the closely related method MLFM [Hoff, 2009]. The $\mathbf{U}$ and $\mathbf{V}$ in both MLFM and GLFM are initialized by principal component analysis (PCA) on the content information. In addition, we also adopt the methods introduced in [Yang *et al.*, 2009a] and [Yang *et al.*, 2009b] for comparison. Those methods can be divided into three groups: link-based methods, content-based methods, link+content based methods.

The link-based methods include: PHITS [Cohn and Chang, 2000], LDA-Link [Erosheva *et al.*, 2004]–an extension of latent Dirichlet allocation (LDA) for link

[4] The two data sets can be downloaded from `http://www.cs.umd.edu/projects/linqs/projects/lbc/index.html`.

analysis, the popularity-based conditional link model (PCL) [Yang *et al.*, 2009b], and the normalized cut (NCUT) for spectral clustering [Shi and Malik, 2000].

The content-based methods include: the probabilistic latent semantic analysis (PLSA) [Hofmann, 1999], LDA-Word, and NCUT respectively with the Gaussian RBF kernel and the probabilistic product (PP) kernel [Jebara *et al.*, 2004].

The link+content based methods include: PHITS-PLSA [Cohn and Hofmann, 2000], LDA-Link-Word [Erosheva *et al.*, 2004], Link-Content-Factorization (LCF) [Zhu *et al.*, 2007], NCUT, PCL-PLSA, PHITS-DC, PCL-DC and C-PLDC [Yang *et al.*, 2009a]. Here PCL-PLSA represents the combination of PCL and PLSA, PHITS-DC represents the PHITS model combined with the discriminative content (DC) model in [Yang *et al.*, 2009a] , PCL-DC represents the PCL model combined with DC, and C-PLDC refers to the combined popularity-driven link model and DC model [Yang *et al.*, 2009a]. Moreover, the setting for t in C-PLDC follows that in [Yang *et al.*, 2009a]. More specifically, C-PLDC($t = 1$) denotes a special case of C-PLDC without popularity modeling [Yang *et al.*, 2009a].

5.3 Illustration

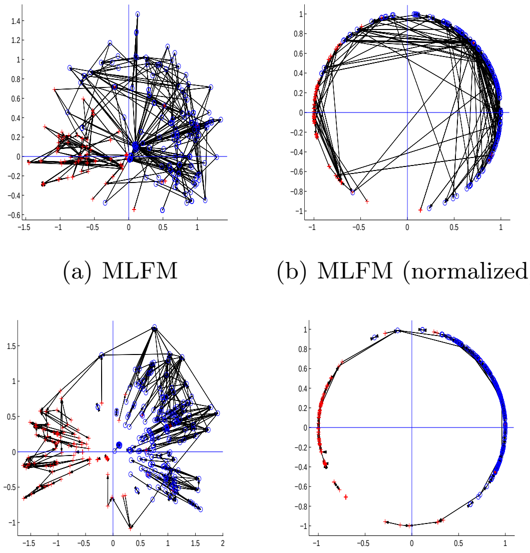
We sample a subset from Cora for illustration. The sampled data set contains two classes. The learned latent representations $\mathbf{U}$ for the data instances are illustrated in Figure 2, where the blue circle and red cross are used to denote the data instances from two different classes respectively, and the (directed) black edges are the citation relationships between the data points. In Figure 2, (a) and (c) show the original learned latent factors of MLFM and GLFM, respectively, (b) and (d) show the corresponding normalized latent factors of MLFM and GLFM, respectively. Here normalization means that we divide the latent factor of each node by its length. Hence, it is clear to see that all the points in (b) and (d) have unit length. Note that for fair comparison all the different subfigures from (a) to (d) are generated automatically by our program with the same parameter settings and initial values.

In (a) and (b) of Figure 2, two instances are more likely to be close if they are connected by or connect to the same instance, which is just the feature of stochastic equivalence. However, there exist many links across the inner part of the circle in (b), which means that two instances linked with each other are not necessarily close in the latent space. This just violates the feature of homophily. Hence, we can conclude that MLFM cannot effectively model homophily in networks.

In (c) and (d) of Figure 2, homophily is obvious since two nodes are close to each other in general if there exists a link between them.

5.4 Convergence Speed

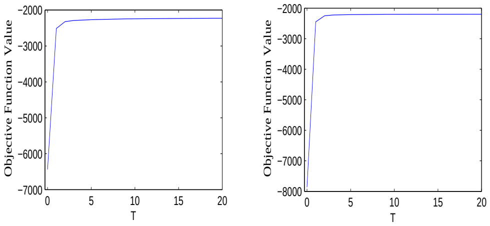
When $D = 20$, the objective function values of GLFM against the iteration number T are plotted in Figure 3, from which we can see that our learning procedure with the MM method for GLFM converges very fast. We set the maximum number of iterations T as $T = 5$ in all our following experiments.



(a) MLFM (b) MLFM (normalized)

(c) GLFM (d) GLFM (normalized)

Figure 2: Illustration of the homophily and stochastic equivalence modeling in networks.



(a) Cora (b) Citeseer

Figure 3: Convergence speed of GLFM.

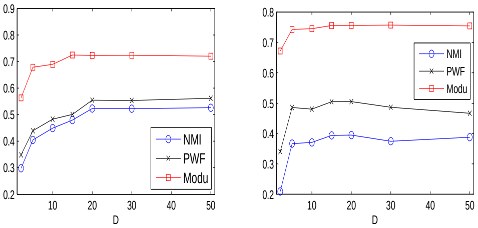
5.5 Accuracy

We compare GLFM with all the baselines introduced in Section 5.2 in terms of NMI, PWF and $Modu$ [Yang *et al.*, 2009a; 2009b]. The results are reported in Table 1, from which we can see that GLFM achieves the best performance on all the data sets for the three criteria. Especially for the Citeseer data set, GLFM dramatically outperforms the second best model. According to the prior knowledge, the paper citation networks are more likely to exhibit homophily because the citations often exist among papers from the same community. This can explain why GLFM can achieve such good performance on these data sets. Hence, GLFM provides a way to model networks which cannot be modeled well by MLFM.

Figure 4 shows the performance of GLFM when D takes different values. We see that GLFM is not sensitive to D as long as D is not too small.

Table 1: Community detection performance on Cora and Citeseer data sets (the best performance is shown in bold face).

		Cora			Citeseer		
	Algorithm	NMI	PWF	$Modu$	NMI	PWF	$Modu$
Link	PHITS	0.0570	0.1894	0.3929	0.0101	0.1773	0.4588
	LDA-Link	0.0762	0.2278	0.2189	0.0356	0.2363	0.2211
	PCL	0.0884	0.2055	0.5903	0.0315	0.1927	0.6436
	NCUT	0.1715	0.2864	0.2701	0.1833	0.3252	0.6577
Content	PLSA	0.2107	0.2864	0.2682	0.0965	0.2298	0.2885
	LDA-Word	0.2310	0.2774	0.2970	0.1342	0.2880	0.3022
	NCUT(RBF kernel)	0.1317	0.2457	0.1839	0.0976	0.2386	0.2133
	NCUT(pp kernel)	0.1804	0.2912	0.2487	0.1986	0.3282	0.4802
Link + Content	PHITS-PLSA	0.3140	0.3526	0.3956	0.1188	0.2596	0.3863
	LDA-Link-Word	0.3587	0.3969	0.4576	0.1920	0.3045	0.5058
	LCF	0.1227	0.2456	0.1664	0.0934	0.2361	0.2011
	NCUT(RBF kernel)	0.2444	0.3062	0.3703	0.1592	0.2957	0.4280
	NCUT(pp kernel)	0.3866	0.4214	0.5158	0.1986	0.3282	0.4802
	PCL-PLSA	0.3900	0.4233	0.5503	0.2207	0.3334	0.5505
	PHITS-DC	0.4359	0.4526	0.6384	0.2062	0.3295	0.6117
	PCL-DC	0.5123	0.5450	0.6976	0.2921	0.3876	0.6857
	C-PLDC(t=1)	0.4294	0.4264	0.5877	0.2303	0.3340	0.5530
	C-PLDC	0.4887	0.4638	0.6160	0.2756	0.3611	0.5582
	MLFM	0.3640	0.3874	0.2325	0.2558	0.3356	0.0089
	GLFM	**0.5229**	**0.5545**	**0.7234**	**0.3951**	**0.5053**	**0.7563**



(a) Cora (b) Citeseer

Figure 4: Sensitivity to the parameter D of GLFM.

6 Conclusion

In this paper, a generalized latent factor model is proposed to model homophily in social networks. A linear-time learning algorithm with convergence guarantee is proposed to learn the parameters. Experimental results on community detection in real-world networks show that our model can effectively model homophily to outperform state-of-the-art methods.

Acknowledgments

Li is supported by a grant from the "project of Arts & Science" of Shanghai Jiao Tong University (No. 10JCY09). Yeung is supported by General Research Fund 622209 from the Research Grants Council of Hong Kong. Zhang is supported by the NSFC (No. 61070239), the 973 Program of China (No. 2010CB327903), the Doctoral Program of Specialized Research Fund of Chinese Universities (No. 20090101120066), and the Fundamental Research Funds for the Central Universities.

References

[Airoldi *et al.*, 2008] Edoardo M. Airoldi, David M. Blei, Stephen E. Fienberg, and Eric P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2008.

[Bartholomew and Knott, 1999] David J. Bartholomew and Martin Knott. *Latent Variable Models and Factor Analysis*. Kendall's Library of Statistics,7, second edition, 1999.

[Cohn and Chang, 2000] David Cohn and Huan Chang. Learning to probabilistically identify authoritative documents. In *ICML*, 2000.

[Cohn and Hofmann, 2000] David A. Cohn and Thomas Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. In *NIPS*, 2000.

[Dempster *et al.*, 1977] A Dempster, N Laird, and D Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[Erosheva *et al.*, 2004] Elena Erosheva, Stephen Fienberg, and John Lafferty. Mixed membership models of scientific publications. In *Proceedings of the National Academy of Sciences*, volume 101, pages 5220–5227, 2004.

[Goldenberg *et al.*, 2009] Anna Goldenberg, Alice X. Zheng, Stephen E. Fienberg, and Edoardo M. Airoldi. A survey of statistical network models. *Foundations and Trends in Machine Learning*, 2:129–233, 2009.

[Hoff *et al.*, 2002] Peter D. Hoff, Adrian E. Raftery, and Mark S. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97(460):1090–1098, 2002.

[Hoff, 2008] Peter D. Hoff. Modeling homophily and stochastic equivalence in symmetric relational data. In *NIPS*, 2008.

[Hoff, 2009] Peter D. Hoff. Multiplicative latent factor models for description and prediction of social networks. *Computational and Mathematical Organization Theory*, 15:261–272, 2009.

[Hofmann, 1999] Thomas Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, 1999.

[Jebara *et al.*, 2004] Tony Jebara, Risi Imre Kondor, and Andrew Howard. Probability product kernels. *Journal of Machine Learning Research*, 5:819–844, 2004.

[Kemp *et al.*, 2006] Charles Kemp, Joshua B. Tenenbaum, Thomas L. Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *AAAI*, 2006.

[Lang *et al.*, 2000] Kenneth Lang, David R. Hunter, and Ilsoon Yang. Optimization transfer using surrogate objective functions. *Journal of Computational and Graphical Statistics*, 9, 2000.

[Li and Yeung, 2009] Wu-Jun Li and Dit-Yan Yeung. Relation regularized matrix factorization. In *IJCAI*, 2009.

[Li *et al.*, 2009a] Wu-Jun Li, Dit-Yan Yeung, and Zhihua Zhang. Probabilistic relational PCA. In *NIPS*, 2009.

[Li *et al.*, 2009b] Wu-Jun Li, Zhihua Zhang, and Dit-Yan Yeung. Latent Wishart processes for relational kernel learning. In *AISTATS*, 2009.

[Nowicki and Snijders, 2001] Krzysztof Nowicki and Tom A. B. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96:1077–1087, 2001.

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

[Wasserman and Faust, 1994] Stanley Wasserman and Katherine Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.

[Yang *et al.*, 2009a] Tianbao Yang, Rong Jin, Yun Chi, and Shenghuo Zhu. A Bayesian framework for community detedtion integrating content and link. In *UAI*, 2009.

[Yang *et al.*, 2009b] Tianbao Yang, Rong Jin, Yun Chi, and Shenghuo Zhu. Combining link and content for community detection: a discriminative approach. In *KDD*, 2009.

[Zhu *et al.*, 2007] Shenghuo Zhu, Kai Yu, Yun Chi, and Yihong Gong. Combining content and link for classification using matrix factorization. In *SIGIR*, 2007.