12-1-2013

Community Detection in Complex Networks

Yaojia Zhu

Yaojia Zhu

Candidate

Computer Science

Department

This dissertation is approved, and it is acceptable in quality and form for publication:

Approved by the Dissertation Committee:

Jared Saia , Chairperson

Cristopher Moore

Stephanie Forrest

Aaron Clauset

Community Detection in Complex Networks

by

Yaojia Zhu

B.E., Shanghai Jiao Tong University, 2004
M.E., Shanghai Jiao Tong University, 2007

DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY
Computer Science

The University of New Mexico

Albuquerque, New Mexico

December, 2013

Dedication

To my wife and my parents, for their support and encouragement they're giving me for graduation.

Acknowledgments

I would like to thank my advisor, Professor Cristopher Moore, for his support on my PhD study, academic research and my life in USA. I would like to thank other committe members: Stephanie Forrest, Jared Saia and Aaron Clauset for their invaluable comments and suggestions on my research work. I would like to thank my colleague, Xiaoran Yan, for his wonderful contributions to our collaborated research projects. I'm grateful to Lise Getoor, Terran Lane, Ben Edwards, Brian Ball, Brian Karrer, Mark Newman and David M. Blei for their helpful conversations on my research projects. I would thank to some others, like Zhang Jiang, Mingyan Xu and Thanaphon Tangchoopong, for their help in these years.

Community Detection in Complex Networks

by

Yaojia Zhu

B.E., Shanghai Jiao Tong University, 2004
M.E., Shanghai Jiao Tong University, 2007

PhD, Computer Science, University of New Mexico, 2013

Abstract

The stochastic block model is a powerful tool for inferring community structure from network topology. However, the simple block model considers community structure as the only underlying attribute for forming the relational interactions among the nodes, this makes it prefer a Poisson degree distribution within each community, while most real-world networks have a heavy-tailed degree distribution. This is essentially because the simple assumption under the traditional block model is not consistent with some real-world circumstances where factors other than the community memberships such as overall popularity also heavily affect the pattern of the relational interactions. The degree-corrected block model can accommodate arbitrary degree distributions within communities by taking nodes' popularity or degree into account. But since it takes the vertex degrees as parameters rather than generating them, it cannot use them to help it classify the vertices, and its natural generalization to directed graphs cannot even use the orientations of the edges.

We developed several variants of the block model with the best of both worlds:

they can use vertex degrees and edge orientations in the classification process, while tolerating heavy-tailed degree distributions within communities. We show that for some networks, including synthetic networks and networks of word adjacencies in English text, these new block models achieve a higher accuracy than either standard or degree-corrected block models.

Another part of my work is to develop even more generalized block models, which incorporates other attributes of the nodes. Many data sets contain rich information about objects, as well as pairwise relations between them. For instance, in networks of websites, scientific papers, patents and other documents, each node has content consisting of a collection of words, as well as hyperlinks or citations to other nodes. In order to perform inference on such data sets, and make predictions and recommendations, it is useful to have models that are able to capture the processes which generate the text at each node as well as the links between them. Our work combines classic ideas in topic modeling with a variant of the mixed-membership block model recently developed in the statistical physics community. The resulting model has the advantage that its parameters, including the mixture of topics of each document and the resulting overlapping communities, can be inferred with a simple and scalable expectation- maximization algorithm.

We test our model on three data sets, performing unsupervised topic classification and link prediction. For both tasks, our model outperforms several existing state-of-the-art methods, achieving higher accuracy with significantly less computation, analyzing a data set with 1.3 million words and 44 thousand links in a few minutes.

Contents

Glossary

K	Number of communities.
N	Number of nodes.
M	Number of links.
ω	Mixing matrix of the stochastic block models.
S_u	Degree-correction parameter, i.e., the overall popularity of node u.
θ_d	Topic mixture of document d.
β_z	Word distribution for topic z.
S_d	Degree-correction parameter, i.e., the overall popularity of document d.
L_d	Length of document d.
W_{dn}	nth word in document d.
$A_{dd'}$	Number of links between documents d and d'.
η_z	Link density for topic z.
SBM	The *stochastic block model* without degree-correction.

DC	The *degree-corrected* block model for undirected networks.
DDC	The *directed degree-corrected* block model.
ODC	The *oriented degree-corrected* block model.
DG	*Degree generation* for degree-corrected block models.
PMTLM	The *Poisson mixed-topic link model*.
PMTLM-DC	The *Poisson mixed-topic link model with degree-correction*.
MCMC	The *Markov Chain Monte Carlo* method.
KL	The *Kernighan-Lin* heuristic.
NMI	Normalized mutual information.

Chapter 1

Introduction

Complex networks are prevalent in the real world and have been widely studied in many different areas including social networks, technological and information networks and biological networks. These networks typically exhibit non-trivial topological features, like heavy-tail degree distributions, high clustering coefficients, assortative or disassortative mixing patterns and so on. One of the key questions in complex network analysis revolves around the identification of hidden community structure or clusters based on the observed topological information. Community structure detection for complex networks provides guidance for further study of these networks. The *stochastic block model* (SBM) [23, 31, 51, 2] is a widely used and highly flexible generative model for community detection in complex networks. As a generative model, SBM formally models the generative process of network topology given the hidden community membership of the nodes.

Traditionally, the block model infers latent group structure from the connection pattern. Nodes of the networks are partitioned into blocks. The distribution of the edges between nodes depends on the blocks to which the nodes belong. Here the block to which a node belongs can be viewed as a hidden attribute of that node. Then the

block assignments can be viewed as random variables which affect the relationships among the nodes, i.e., the edges connecting those nodes. The underlying assumption of the stochastic block model is that the nodes belonging to the same block are stochastically equivalent in the sense that the probabilities of the relationships with all other nodes are the same for all nodes in the same block.

However, besides the hidden attribute (i.e., the latent group membership) node other attributes may also affect its relationships with other nodes. One such attribute is node degree. Block models which don't incorporate the information of node degrees, group nodes with similar degrees. This is not true for many real-world networks, where nodes with highly skewed degrees can be in the same block. A natural extension to handle this is to think about both block assignment and degree as covariates of the relational structure of the data. Recently, Karrer and Newman [33] developed a generalized block model called the degree-corrected block model, to incorporate node degrees. This elaborate block model assumes that the expected number of links between two nodes are proportional to their degrees.

As an extension of that model, we developed a degree-corrected block model called the directed degree-corrected block model (DDC) for directed graphs. We found that DDC works well for some networks but it cannot use the edge orientation information for those networks where the inter-community connections are highly directed, i.e., most of the edges between two blocks direct from one to another. To fill this gap, we developed another model called the oriented degree-corrected block model which first generates the undirected network and then generates the edge orientations.

Non-degree-corrected block models assume similar degrees within communities. On the other hand, degree-corrected models prefer highly skewed degree distributions in each community. What if we want both, Poisson degree distribution in block 1 and power-law degree distribution in block 2? Or more generally, how can we use domain knowledge about the degree distributions of the community structure we are

looking for? To address this, we introduced degree generation (DG) for the degree-corrected block models to take advantage of the domain knowledge about the degree distributions in each community. DG first generates average degrees according to the domain knowledge and then applies the degreee-corrected block mdoel to discover the communities.

Besides the hidden community membership and the degree, other node attributes may also provide valuable information for community detection. In that case, an even more generalized block model should incorporate them too. For example, in document citation networks, or online web pages, we have text for each document besides the links for document pairs. Here the links are the citation relationships or hyperlinks. We developed the Poisson mixed-topic link models (PMTLM) which are able to detect community structures based on both text and links. PMTLM is a combination of the mixed-membership stochastic block model, which we call the "Ball-Karrer-Newman" (BKN) model [7], and the probabilitic latent semantic analysis (PLSA) [30] model.

As described above, this dissertation focuses on generalizing stochastic block models for community detection in complex networks, including the degree-corrected block models for directed networks, degree generation and the mixed-topic link models which analyses both text and links. Contributions include:

1. We extended the Karrer-Newman degree-corrected block model [33] to directed networks. New models include directed degree-corrected and oriented degree-corrected block models. We analyzed the strengths and weaknesses of the various versions of the block model and provided guidance on how to apply these models for community detection. We introduced degree generation for degree-corrected block models. As a generative model, degree generation generates node expected degrees, allows the model to capture the interaction between the degree distribution and the community structure. In particular, degree

generation automatically strikes a balance between allowing vertices of different degrees to coexist in the same community on the one hand, and using vertex degrees to separate vertices into communities on the other. We believe our work provides valuable theoretical and experimental support for guiding the application of these models to detect different kinds of community structures. Furthermore, such analysis and understanding is vital for developing new block models on other specific networks.

2. The mixed-topic link model explores a new way to incorporate node rich information into link-based stochastic block models. The new model combines a classic content-based topic model, which is called the probabilitic latent semantic analysis (PLSA) [30] and the BKN [7] mixed membership block model. We developed a simple and highly efficient inference algorithm, which is linear in the size of the data set. Experiments on real-world data sets show that the new model achieves the state-of-the-art performance on both document classification and link prediciton tasks. Some related issues like balancing content and links, applying local search optimization and predicting the missing links are all addressed.

The following chapters are organized as follows: Chapter 2 reviews the background and the related work. Chapter 3 presents degree-corrected block models for directed networks and degree-generation. Chapter 4 discusses the mixed-topic link models. Chapter 5 gives the conclusion of the current work and Chapter 6 talks about some future directions.

Chapter 2

Background

2.1 Community detection in complex networks

In many real-world networks, vertices can be divided into *communities* based on their connections. Social networks can be forged by daily interactions like karate training [58], the blogosphere contains groups of linked blogs with similar political views [1], words can be tagged as different parts of speech based on their adjacencies in large texts [48], and so on. One of the key questions in complex network analysis revolves around the identification of the hidden community structure.

Community structures can be very different in real-world networks. Communities range from assortative clumps, where vertices preferentially attach to others of the same type, to *functional* communities of vertices that connect to the rest of the network in similar ways, such as groups of predators in a food web that feed on similar prey [5, 42]. Networks may exhibit hierarchical community organization, in which communities are associated to levels or scales and communities on higher levels contain sub-communities on lower levels. By contrast, a flat structure assumes that all communities are at the same level. Communities may overlap, in which nodes

can involve several communities simultaneously. By contrast, a non-overlapping community structure only consists of disjoint communities. Understanding various community structures, and their relations to the functional roles of vertices and edges, is crucial to understanding network data and providing a very helpful guidance for further study of those networks.

2.2 Stochastic block modeling

The *stochastic block model* (SBM) [23, 31, 51, 2] is a popular and highly flexible generative model for community detection. It partitions the vertices into communities or *blocks*, where vertices belonging to the same block are *stochastically equivalent* [53] in the sense that the probabilities of a connection with all other vertices are the same for all vertices in the same block. With this rather general definition of community, block models can capture many types of community structure, including assortative, disassortative, and satellite communities and mixtures of them [45, 46, 43, 42, 21, 20].

The purpose of block modeling [35, 54] is to partition the node set into subsets–*blocks*, where nodes belong to the same block are structurally equivalent [35]. In such a way we hope the block structure and the edge pattern between the blocks can capture the main structural features of the graph. Stochastic modeling [32] is another approach to relational data analysis. These two approaches have their own strengths and weaknesses, but they are strongly complementary [31]. Fienberg and Wasserman [23] and Holland, Laskey and Leinhardt [31] extended the concept of block modeling to a stochastic version, which is the integration of those two approaches that hopefully can overcome the limitations of each while creating a statistical methodology that is consistent, effective and broadly applicable. Under such a model, the nodes belong to the same block are stochastically equivalent [53] in the sense that the probabilities of the relationships with all other nodes are the

same for all nodes in the same block. Based on the concept of stochastic equivalence, stochastic block modeling can detect group memberships of the interact nodes for both assortative and disassortative mixing [46]. Here assortativity is a bias in favor of connections between nodes with similar characteristics, and disassortativity is a bias in favor of connections between dissimilar nodes. In social networks, for example, individuals commonly choose to associate with others of similar features as themselves, which is also known as *homophily* [39, 24]. In networks of sexual contact, the mixing is disassortative by gender - most partnerships are between individuals of opposite sex. Food webs also present disassortative mixing, as species on the same trophic level mostly interact with species on other levels [25, 42].

As a generative model, the block model can be used to generate stochastic networks if the blocks are known. In this project, we assume the blocks are unobserved (latent), and we establish block models for detecting those latent blocks. This is sometimes called a posteriori block modeling [53, 6, 33, 55]. Traditionally, the block model infers latent group structure from connection patterns. Nodes of the network are partitioned into blocks. The distribution of the edges between nodes is dependent on the blocks to which the nodes belong. Here the block to which a node belongs can be viewed as a hidden attribute of that node. Then the block memberships can be viewed as random variables which affect the relationships among the nodes, i.e., the edges connecting those nodes. The underlying assumption of the stochastic block model is that the nodes belonging to the same block are stochastically equivalent in the sense that the probabilities of the relationships with all other nodes are the same for all nodes in the same block.

2.2.1 Generative model

We have the variable Y, which can be discrete values or labels for classification problems, or continuous values for regression problems. We also have the observed data

X. What we are interested in is the conditional probability of the Y given the data X, i.e., $p(Y|X)$. There are two categories of models to calculate this conditional probability distribution: discriminative models and generative models. Discriminative models work on the conditional probability $p(Y|X)$. Some well known examples include linear regression, logistic regression, support vector machines and neural networks. Generative models focus on the joint probability $p(XY)$, and the conditional probability $p(Y|X)$ can be obtained according to Bayes' rule,

$$p(Y|X) = \frac{p(XY)}{p(X)} = \frac{p(X|Y)p(Y)}{p(X)}\,. \tag{2.1}$$

Here $p(Y|X)$ and $p(Y)$ are the posterior and prior distributions of Y respectively, and $p(X|Y)$ is called the likelihood function. $p(X)$ is the normalization term.

The likelihood function $p(X|Y)$ is described as a generative proceess by which the observed data X is generated given Y. Consequently, *a generative model can only learn from the observed data that it is required to generate.* This principle guides us the development of the models described in Chapter 3 and Chapter 4.

Given the observed data X, the maximum a posteriori (MAP) estimate is the mode of the posterior distribution, i.e., the value of Y that maximizes the posterior distribution $p(Y|X)$,

$$\hat{y}_{\mathrm{MAP}} = \underset{Y}{\operatorname{argmax}}\, p(Y|X)\,. \tag{2.2}$$

The maximum likelihood estimate (MLE) is the mode of the likelihood function,

$$\hat{y}_{\mathrm{MLE}} = \underset{Y}{\operatorname{argmax}}\, p(X|Y)\,. \tag{2.3}$$

According to (2.1), $\hat{y}_{\mathrm{MAP}} = \hat{y}_{\mathrm{MLE}}$ holds when $p(Y)$ is a uniform prior.

Some well-known examples of generative models include the Gaussian mixture models, hidden Markov models, Naive Bayes classifiers, latent Dirichlet allocation and so on. The stochastic block models are well-known generative models for com-

munity detection. In stochastic block models, the observed data X is the network, and Y represents the community memberships of the nodes we want to infer.

2.2.2 Bernoulli stochastic block model

This is the most basic version of the stochastic block model. Let's consider an undirected network, which is denoted as a simple graph G, including N nodes and M links. We want to infer a N-dimensional vector $g = \{g_1, g_2 \ldots, g_N\}$, in which $g_u \in \{1, \ldots, K\}$ gives the block membership of node u. The number of topics, which is denoted by K, is assumed to be known. The model has a $K \times K$ symmetric matrix p, in which p_{rs} denotes the connection probability between block r and block s. Then the posteriori distribution of the block memberships g is the following,

$$P(g|G,p) = \frac{P(G|g,p)P(g)}{P(G)} \propto P(G|g,p)P(g)\,. \tag{2.4}$$

Given the p parameters, the Bernoulli stochastic block model gives the following likelihood function,

$$\begin{aligned} P(G|g,p) &= \prod_{(u,v)\in E} p_{g_u g_v} \times \prod_{(u,v)\notin E} (1 - p_{g_u g_v}) \\ &= \prod_{u<v} (p_{g_u g_v})^{A_{uv}} (1 - p_{g_u g_v})^{1-A_{uv}}\,. \end{aligned} \tag{2.5}$$

The model assumes that each edge is generated independently conditioned on the block memberships. Each entry A_{uv} of the adjacency matrix is then Bernoulli-distributed, where the probability that $A_{uv} = 1$ depends solely on the block memberships g_u, g_v of its endpoints.

2.2.3 Poisson stochastic block model

In the original stochastic block model, the entries A_{uv} of the adjacency matrix are independent and Bernoulli-distributed, with $P(A_{uv} = 1) = p_{g_u,g_v}$. Here g_u is the

block to which u belongs, where p is a $K \times K$ matrix. Karrer and Newman [33] consider random multigraphs where the A_{uv} are independent and Poisson-distributed, namely, $A_{uv} \sim \text{Poi}(\omega_{g_u,g_v})$, and ω replaces p as the mixing matrix.

Thus, ignoring self-loops, the likelihood function for generating an undirected graph G given the group assignment g and the mixing matrix ω is the following:

$$P(G \mid g, \omega) = \prod_{u<v} \frac{(\omega_{g_u g_v})^{A_{uv}}}{A_{uv}!} \exp(-\omega_{g_u g_v}) \,. \tag{2.6}$$

Ignoring the constant $\sum_{u<v} \log A_{uv}!$, the log-likelihood is

$$\log P(G \mid g, \omega) = \sum_{u<v} \left(A_{uv} \log \omega_{g_u g_v} - \omega_{g_u g_v}\right) \,. \tag{2.7}$$

For each pair of blocks r, s, the maximum likelihood estimate (MLE) for ω_{rs} is

$$\hat{\omega}_{rs} = \frac{m_{rs}}{n_r n_s} \,. \tag{2.8}$$

Here m_{rs} is the number of edges between group r and s if $r \neq s$, and twice the number of edges within group r when $r = s$. The number of nodes in group r is denoted as n_r.

Substituting the ω in (2.7) with the maximum likelihood estimates (MLEs) gives the profile log-likelihood [10]

$$\log P(G \mid g) = \frac{1}{2} \sum_{rs} m_{rs} \log \frac{m_{rs}}{n_r n_s} \,. \tag{2.9}$$

The factor $\frac{1}{2}$ can be ignored for maximizing the value of the likelihood function. That is exactly what we saw in [33]. For future reference, we call this model the Stochastic Block Model (SBM).

2.2.4 Degree-corrected block model for undirected graphs

In the stochastic block model defined in the previous section, every pair of vertices in a given pair of blocks are connected with the same probability. Thus, for large n the

degree distribution within each block is Poisson. As a consequence, vertices with very different degrees are unlikely to be in the same block. This leads to problems with modeling real networks, which often have heavy-tailed degree distributions within each community. For instance, both liberal and conservative political blogs range from high-degree "leaders" to low-degree "followers" [1].

To avoid this effect, and allow degree inhomogeneity within blocks, there is a long history of generative models where the probability of an edge depends on vertex attributes as well as their block memberships. A particularly elegant variant is the *degree-corrected* block model (DC) developed by Karrer and Newman [33].

The DC model generates random multigraphs where the A_{uv} are independent and Poisson-distributed,

$$A_{uv} \sim \mathrm{Poi}(S_u S_v \omega_{g_u, g_v}) \,, \tag{2.10}$$

here S_u is an overall propensity for u to connect to other vertices. Note that since the A_{uv} are independent, the degrees d_u will vary somewhat around their expectations; however, the resulting model is much simpler to analyze than one that controls the degree of each vertex exactly. The DC model prefers high entropy of the degrees within community, thus encourages hetorogeneous degree distributions in each community.

To remove the obvious symmetry where we multiply the S parameters by a constant C and divide ω by C^2, a normalization constraint $\sum_{u:g_u=r} S_u = \kappa_r$ is imposed for each block r, where $\kappa_r = \sum_{u:g_u=r} d_u$ is the total degree of the vertices in block r. Under these constraints, the MLEs for the S and ω parameters are then

$$\hat{S}_u = d_u, \quad \hat{\omega}_{rs} = \frac{m_{rs}}{\kappa_r \kappa_s} \,, \tag{2.11}$$

where m_{rs} is the same notation in (2.8). Substituting these MLEs for S and ω then

gives the profile log-likelihood

$$\log P(G \mid g) = \frac{1}{2} \sum_{r,s=1}^{K} m_{rs} \log \frac{m_{rs}}{\kappa_r \kappa_s} . \tag{2.12}$$

Here κ_r is the total number of degrees in group r.

2.2.5 Mixed-membership block model

Mixed-membership block models are designed for networks in which the communities are overlapped, i.e., some nodes may belong to more than one community. This is very common in real-world networks. In social networks, people are associated with different groups with multiple roles at the same time; in document citation networks, or online web pages, documents may involve multiple topics. A well known mixed-membership block model is the *Mixed Membership Stochastic Blockmodel* (MMSB), which is presented in [2]. Like the Bernoulli stochastic block model, MMSB also assumes Bernoulli distributions for each link. Another mixed-membership block model, which is proposed in [33], used Poisson distributions for the number of links between each pair of nodes. We call this Poisson model the *Ball-Karrer-Newman* (BKN) model, and we used this model in our *Poisson Mixed-Topic Link Model* (PMTLM), which is described in Chapter 4.

In block models without overlapping communities, each node u belongs to only one block, which is denoted as $g_u \in \{1, \ldots, K\}$. In mixed-membership block models, each node u is associated with a K-dimensional mixed membership vector θ_u. For any block $z \in \{1, \ldots, K\}$, θ_{uz} quantifies the extent to which node u belongs to block z.

In MMSB, these θ parameters are hidden variables drawn from a Dirichlet prior. The posterior distribution of θ is calculated approximately using the variational approach, which is quite expensive. In BKN, the θ parameters are estimated to

maximize the likelihood using a highly efficient EM algorithm. More discussion about the difference of these two models is presented in Chapter 4.

2.3 Topic modeling

Let's consider networks in which each node has rich attributes. This kind of data is very common in the real world. Like in document citation networks or online web pages, besides the pairwise relationship, which are the citation relationships or hyperlinks, each document contains a bunch of words. In these networks, each document is a node, and the communities are the topics. The node attributes, i.e., the document content, provides valuable information for determining the document topics. Thus, a model which uses both the content (first order) and link (second order) information to detect document topics is appealing. Stochastic block modeling, which is described in the previous section, provides a powerful tool to detect the community structures by using the network topology, i.e., the link information. On the other hand, content-based topic models [30, 11], which are well studied in topic modeling research, give a formal way to use the content informaiton.

Normally topic models assume that each document belongs to multiple topics, which is the same assumption made in mixed membership block models. We will see in Chapter 4 that the mixed-membership block model and the probabilistic topic models are naturally compatable with each other, giving us a nice model to analyse both content- and link-information efficiently.

In models such as Probabilistic Latent Semantic Analysis (PLSA) [30] and Latent Dirichlet Allocation (LDA) [11], each document d has a K-dimensional mixture θ_d of topics. Each topic z corresponds in turn to a probability distribution over words, which is denoted as β_z, and each word in d is generated independently from the resulting mixture of distributions.

Consider a network of N documents. Each document d has a fixed length L_d, and consists of a string of words $w_{d\ell}$ for $1 \leq \ell \leq L_d$, where $1 \leq w_{d\ell} \leq W$ where W is the number of distinct words. In the PLSA model [30], the generative process for the content is described as follows. For each document $1 \leq d \leq N$ and each $1 \leq \ell \leq L_d$, we independently choose a topic $z = z_{d\ell} \sim \text{Multi}(\theta_d)$, and choose the word $w_{d\ell} \sim \text{Multi}(\beta_z)$. Thus the total probability that $w_{d\ell}$ is a given word w is

$$\Pr[w_{d\ell} = w] = \sum_{z=1}^{K} \theta_{dz}\beta_{zw} \,. \tag{2.13}$$

Like in those stochastic block models, here we also assume that the number of topics K is fixed. The distributions β_z and θ_d are parameters to be inferred.

In LDA [11], the topic mixtures θ are hidden variables and further drawn from a Dirichlet prior. In some variants, a Dirichlet prior is imposed on β variables as well.

Chapter 3

Degree-correction and degree-generation

The work described in this chapter are published in [61] under the supervision of Prof. Cristopher Moore. I collaborated with Xiaoran Yan. Xiaoran helped with the Bayesian estimation for degree-generation models, which is described in Appendix C.

Degree-corrected stochastic block models are powerful tools for dealing with networks with inhomogeneous degree distributions. However, since degree-corrected models are given the vertex degrees as parameters and are under no obligation to explain them, they cannot use degrees to help them classify vertices. As described in Section 2.2.1, a generative model can only learn from the data that it is required to generate. For this reason, the DC model may actually fail to recognize communities that differ significantly in their degree distributions. Thus we have two extremes: the SBM separates vertices by degree even when it shouldn't, and the DC model fails to do so even when it should. Here the SBM and the DC models [33] are the previous work described in Section 2.2.3 and Section 2.2.4.

We have a similar problem for directed graphs. The natural generalization of the DC model, the *directed degree-corrected* (DDC) block model, which is described in the following section, has two parameters for each vertex: the expected in-degree and out-degree. But this model cannot even take advantage of edge orientations. For instance, in English adjectives usually precede nouns but rarely vice versa. Thus the ratio of each vertex's in- and out-degree is strongly indicative of its block membership if part of speech is what the blocks represent. But the DDC model takes these degrees as parameters, so it is unable to use this part of the data to classify words according to their parts of speech.

In the following section, we propose a new degree-corrected block model, which combines the strengths of the degree-corrected and uncorrected block models. The *oriented degree-corrected* (ODC) block model is able to utilize the edge orientations for community detection by only correcting the total degrees. We show that for networks with strongly asymmetric behavior between communities, including synthetic networks and some real-world networks, ODC achieves a higher accuracy than SBM or DDC.

We also propose the *degree-generated* (DG) block model, which treats the expected degree of each vertex as generated from a prior distribution in each block, such as a power law whose exponent varies from one community to another. By including the probability of these degrees in the likelihood of a given block assignment, the DG model captures the interaction between the degree distribution and the community structure. In particular, it automatically strikes a balance between allowing vertices of different degrees to coexist in the same community on the one hand, and using vertex degrees to separate vertices into communities on the other. Our experiments show that DG works especially well in networks where communities have highly inhomogeneous degree distributions, but where the degree distributions differ significantly between communities. In some cases, DG has a further advantage

in faster convergence as it reshapes the parameter space, providing the algorithm a shortcut to the correct community structure.

These new variants of the block model give us the best of both worlds. They can tolerate heavy-tailed degree distributions within communities, but can also use degrees and edge orientations to help classify the vertices. In addition to their performance on real and synthetic networks, our models illustrate a valuable point about generative models and statistical inference: when inferring the structure of a network, you can only use the information that you try to generate.

We test our models on three word adjacency networks in Section 3.3. Our goal is not to do part-of-speech tagging. There's a huge literature on that, and we don't come close to the state of the art. Our motivation in looking at word networks is simply to find a class of networks with directed and disassortative structure, in order to understand the strengths and weaknesses of the various versions of the block model.

3.1 Directed and oriented degree-corrected block models

Throughout, we use N and M to denote the number of vertices and edges respectively, and K to denote the number of blocks. The problem of determining K is a crucial model selection problem. In some cases, we can use prior domain knowledge, such as the number of different parts of speech, or the number of different factions into which a network split over time. In the absence of such knowledge, a variety of methods have been proposed; in particular, we could compute the likelihood of our various models with different values of K, and apply a suitable penalty term as in the AIC [3] or BIC [49] to discourage overfitting. We leave this to our future work

and assume that K is given here.

The natural extension of DC to directed networks, which we call the *directed degree-corrected* (DDC) block model, has two parameters $S_u^{\text{out}}, S_u^{\text{in}}$ for each vertex. The number of directed edges from u to v is again Poisson-distributed,

$$A_{uv} \sim \text{Poi}(S_u^{\text{out}} S_v^{\text{in}} \omega_{g_u,g_v})\,. \tag{3.1}$$

We impose the constraints

$$\sum_{u:g_u=r} S_u^{\text{out}} = \kappa_r^{\text{out}}, \quad \sum_{u:g_u=r} S_u^{\text{in}} = \kappa_r^{\text{in}} \tag{3.2}$$

for each block r, where $\kappa_r^{\text{out}} = \sum_{u:g_u=r} d_u^{\text{out}}$ and $\kappa_r^{\text{in}} = \sum_{u:g_u=r} d_u^{\text{in}}$ denote the total out- and in-degree of block r. As before, let m_{rs} denote the number of directed edges from block r to block s. Then the likelihood is

$$\begin{aligned} P(G \,|\, S, \omega, g) &= \prod_{uv} \frac{\left(S_u^{\text{out}}\, S_v^{\text{in}}\, \omega_{g_u g_v}\right)^{A_{uv}}}{A_{uv}!} \exp(-S_u^{\text{out}}\, S_v^{\text{in}}\, \omega_{g_u g_v}) \\ &= \frac{\prod_u (S_u^{\text{out}})^{d_u^{\text{out}}} (S_u^{\text{in}})^{d_u^{\text{in}}} \prod_{rs} \omega_{rs}^{m_{rs}} \exp(-\kappa_r^{\text{out}} \kappa_s^{\text{in}} \omega_{rs})}{\prod_{uv} A_{uv}!}\,, \end{aligned} \tag{3.3}$$

Ignoring the constant $\sum_{uv} \log A_{uv}!$, the log-likelihood is

$$\begin{aligned} \log P(G \,|\, S, \omega, g) = &\sum_u (d_u^{\text{out}} \log S_u^{\text{out}} + d_u^{\text{in}} \log S_u^{\text{in}}) \\ &+ \sum_{rs} (m_{rs} \log \omega_{rs} - \kappa_r^{\text{out}} \kappa_s^{\text{in}} \omega_{rs})\,. \end{aligned} \tag{3.4}$$

The MLEs for the parameters (see Appendix A) are

$$\hat{S}_u^{\text{out}} = d_u^{\text{out}}\,, \quad \hat{S}_u^{\text{in}} = d_u^{\text{in}}\,, \quad \hat{\omega}_{rs} = \frac{m_{rs}}{\kappa_r^{\text{out}} \kappa_s^{\text{in}}}\,. \tag{3.5}$$

Ignoring constants again and substituting these MLEs give

$$\log P(G \,|\, g) = \sum_{r,s=1}^{K} m_{rs} \log \frac{m_{rs}}{\kappa_r^{\text{out}} \kappa_s^{\text{in}}}\,. \tag{3.6}$$

In the DDC model, the expected in- and out-degrees of each vertex are completely specified by the S parameters. Thus DDC allows vertices with arbitrary degrees to fit comfortably together in the same block. On the other hand, since the degrees are given as parameters, rather than as data that the model must generate and explain, DDC cannot use them to infer vertex labels. Indeed, it cannot even take advantage of the orientations of the edges, as shown below by its poor performance on networks with strongly asymmetric community structure.

To deal with this, we present a partially degree-corrected block model capable of taking advantage of edge orientations, which we call the *oriented degree-corrected* (ODC) block model. Following the maxim that we can only use the information that we try to generate, we correct only for the total degrees of the vertices, and generate the edges' orientations.

Let $\bar{G}$ denote the undirected version of a directed graph G, i.e., the multigraph resulting from erasing the arrows for each edge. Its adjacency matrix is $\bar{A}_{uv} = A_{uv} + A_{vu}$, so (for instance) $\bar{G}$ has two edges between u and v if G had one pointing in each direction. The ODC model can be thought of as generating $\bar{G}$ according to the undirected degree-corrected model, and then choosing the orientation of each edge according to another matrix ρ_{rs}, where an edge (u, v) is oriented from u to v with probability ρ_{g_u,g_v}. Thus the total log-likelihood is

$$\log P(G \,|\, S, \omega, \rho, g) = \log P(\bar{G} \,|\, S, \omega, g) + \log P(G \,|\, \bar{G}, \rho, g)\,. \tag{3.7}$$

Writing $\bar{m}_{rs} = m_{rs} + m_{sr}$ and $\kappa_r = \kappa_r^{\text{in}} + \kappa_r^{\text{out}}$, we can set S_u and ω_{rs} for the undirected model to their MLEs as in equation 2.11, giving

$$\log P(\bar{G} \,|\, g) = \frac{1}{2} \sum_{r,s=1}^{K} \bar{m}_{rs} \log \frac{\bar{m}_{rs}}{\kappa_r \kappa_s}\,. \tag{3.8}$$

The orientation term is

$$\log P(G \,|\, \bar{G}, \rho, g) = \sum_{r,s=1}^{K} m_{rs} \log \rho_{rs}\,. \tag{3.9}$$

For each r, s we have $\rho_{rs} + \rho_{sr} = 1$, and the MLEs for ρ are

$$\hat{\rho}_{rs} = m_{rs}/\bar{m}_{rs}\,. \tag{3.10}$$

Note that $\hat{\rho}_{rr} = 1/2$ for any r. Substituting the MLEs for ρ and combining (3.8) with (3.9) gives the log-likelihood for the ODC model as follows

$$\log P(G \,|\, g) = \sum_{r,s=1}^{K} m_{rs} \log \frac{m_{rs}}{\kappa_r \kappa_s}\,. \tag{3.11}$$

In order to understand ODC better, we analyze the edge orientation term (3.9) more carefully. Substituting the MLEs for ρ in (3.9) gives

$$\begin{aligned}
\log P(G|\bar{G}, g) &= \frac{1}{2} \sum_{rs} \left(m_{rs} \log \hat{\rho}_{rs} + m_{sr} \log \hat{\rho}_{sr}\right) \\
&= \frac{1}{2} \sum_{r \neq s} \bar{m}_{rs} \left(\hat{\rho}_{rs} \log \hat{\rho}_{rs} + \hat{\rho}_{sr} \log \hat{\rho}_{sr}\right) + \sum_{r} m_{rr} \log \hat{\rho}_{rr} \\
&= -\sum_{r<s} \bar{m}_{rs} \tau(\hat{\rho}_{rs}) - (\log 2) \sum_{r} m_{rr}\,.
\end{aligned} \tag{3.12}$$

Here $\tau(x) = -x \log(x) - (1-x)\log(1-x)$ is the entropy function. The total number of inter-block edges is $\sum_{r<s} \bar{m}_{rs}$, and the total number of intra-block edges is $\sum_r m_{rr}$.

Examining (3.12), we see that the edge orientation term prefers highly directed inter-block connections, i.e., such that $\hat{\rho}_{rs}$ are near 0 or 1, so that $\tau(\hat{\rho}_{rs})$ is minimized. However, as $\tau(\hat{\rho}_{rs}) \leq \log 2$, it also prefers disassortative structures, in which the number of intra-block edges m_{rr} is as small as possible; it has no basis on which to orient these edges, so they contribute a negative term to the log-likelihood.

Thus, while ODC can detect assortative structures due to the undirected term (3.8), and may do better than DC or DDC if the connections between blocks are highly directed (for instance, if there are three blocks, and all inter-block connections are oriented from the "lower" block to the "higher" one), it performs best in disassortative networks with highly-directed connections between blocks, so that the

orientation of most edges is determined by the block assignment of their endpoints. We will see an example of this in a real-world network in Section 3.3.2.

We note that we could reduce ODC's preference for disassortative structure by simply ignoring the second term in (3.12). This would correspond to a generative model where inter-block edges are directed, but intra-block edges are undirected. We have not pursued this.

We can also view ODC as a special case of DDC, where we add the constraint $S_u^{\text{in}} = S_u^{\text{out}}$ for all vertices u (see Appendix B). Moreover, if we set $S_u = 1$ for all u, we obtain the original block model, or rather its Poisson multigraph version where each A_{uv} is Poisson-distributed with mean ω_{g_u,g_v}. Thus SBM $\leq$ ODC $\leq$ DDC, where $A \leq B$ means that model A is a special case of model B, or that B is an elaboration of A (see Figure 3.1). We will see later that since it is forced to explain edge orientations, ODC performs better on some networks than either SBM or DDC.

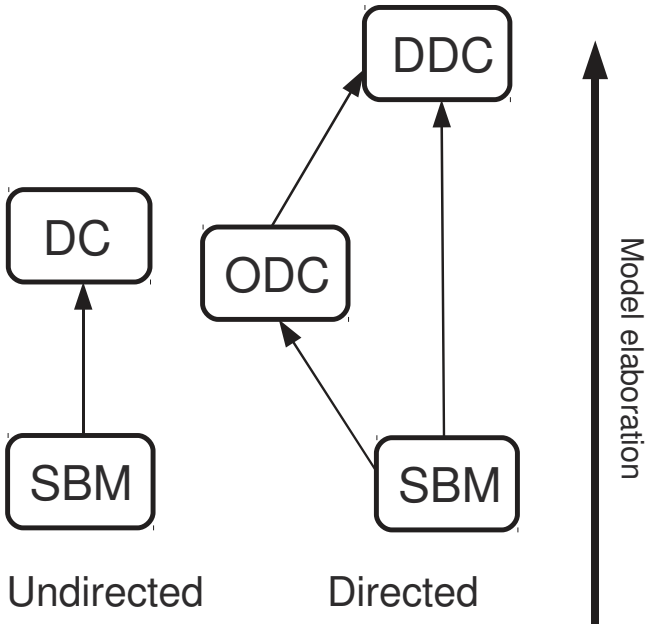


Figure 3.1: Relationship of the models.

3.2 Degree-generated block models

Another way to utilize vertex degrees for community detection is to require the model to generate them, according to some prior degree distribution derived from domain knowledge. For instance, many real-world networks have a power-law degree distribution, but with parameters (such as the exponent, minimum degree, or leading constant) that vary from community to community. In that case, the degree of a vertex gives us a clue as to its block membership. This yields our proposed *degree-generated* (DG) block models. They can tolerate heavy-tailed degree distributions within communities, but can also use degrees and edge orientations to help classify the vertices.

In a DG model, we first generate the S parameters of one of the degree-corrected block models discussed above, i.e., the expected vertex degrees, and then use them to generate a random multigraph. Specifically, each S_u is generated independently according to some distribution whose parameters ψ depend on the block g_u to which u belongs. Thus DG is a hierarchical model, which extends the previous degree-corrected block models by adding a degree generation stage on top, treating the Ss as generated by the block assignment g and the parameters ψ rather than as parameters.

We can apply this approach to the undirected, directed, or oriented versions of the degree-corrected model; at the risk of drowning the reader in acronyms, we denote these DG-DC, DG-DDC, and DG-ODC. In each case, the total log-likelihood of a graph G is

$$\log P(G \mid \psi, \omega, g) = \log \int \mathrm{d}S \, P(G \mid S, \omega, g) \, P(S \mid \psi, g) \,, \tag{3.13}$$

where

$$P(S \mid \psi, g) = \prod_{u=1}^{N} P(S_u \mid \psi_{g_u}) \,. \tag{3.14}$$

For the directed models, we use S_u as a shorthand for S_u^{in} and S_u^{out}.

As in many hierarchical models, computing this integral appears to be difficult, except when $P(S \mid \psi)$ has the form of a conjugate prior such as the Gamma distribution (see Appendix C). We approximate it with a point estimate by assuming that it is dominated by the most-likely value of S,

$$\log P(G \mid \psi, \omega, g) \approx \log P(G \mid \hat{S}, \omega, g) + \log P(\hat{S} \mid \psi, g)\,. \tag{3.15}$$

However, even determining $\hat{S}$ is challenging when $P(S \mid \psi)$ is, say, a power law with a minimum-degree cutoff. Thus we make a further approximation, setting $\hat{S}$ just by maximizing the block model term $\log P(G \mid \hat{S}, \omega, g)$ as we did before, using (3.5) or the analogous equations for the DC or ODC. In essence, these approximations treat $P(\hat{S} \mid \psi, g)$ as a penalty term, imposing a prior on the degree distribution of each community with hyperparameters ψ. This leads to community structures that might not be as good a fit to the edges, but compensate with a much better fit to the degrees.

We can either treat the degree-generating parameters ψ as fixed—say, as predicted by a theoretical model of network growth [4, 9, 41]—or infer them by finding the $\hat{\psi}$ that maximizes $P(\hat{S} \mid \psi)$. For instance, suppose the S_u in block $g_u = r$ are distributed as a continuous power law with a lower cutoff $S_{\min,r}$. Specifically, let the parameters in each block r be $\psi_r = (\alpha_r, \beta_r, S_{\min,r})$, and let

$$P(S_u \mid \psi_r) = \begin{cases} \beta_r & S_u = 0 \\ 0 & 0 < S_u < S_{\min,r} \\ \frac{(1-\beta_r)(\alpha-1)}{S_{\min,r}} \left(\frac{S_u}{S_{\min,r}}\right)^{-\alpha_r} & S_u \geq S_{\min,r}\,. \end{cases} \tag{3.16}$$

In the directed case, we have $\psi_r^{\text{in}} = (\alpha_r^{\text{in}}, \beta_r^{\text{in}}, S_{\min,r}^{\text{in}})$ and $\psi_r^{\text{out}} = (\alpha_r^{\text{out}}, \beta_r^{\text{out}}, S_{\min,r}^{\text{out}})$. Allowing β_r^{out} to be nonzero, for instance, lets us directly include vertices with no outgoing neighbors; we find this useful in some networks. Alternately, we can choose

$(S_u^{\text{in}}, S_u^{\text{out}})$ from some joint distribution, allowing in- and out-degrees to be correlated in various ways.

We fix $S_{\min,r} = 1$. Given the degrees and the block assignment, let $Y_r = \{u : g_u = r \text{ and } S_u \neq 0\}$, and let $y_r = |Y_r|$. The MLE for α_r is [15]

$$\hat{\alpha}_r = 1 + \frac{y_r}{\sum_{u \in Y_r} \ln S_u}\,. \tag{3.17}$$

The MLE for $\hat{\beta}_r$ is simply the fraction of vertices in block r with degree zero.

3.3 Experimental Results

3.3.1 Experiments on synthetic networks

In order to understand under what circumstances our models out-perform previous variants of the block model, we performed experiments on synthetic networks, varying the degree distributions in communities, the degree of directedness between communities, and so on.

First, we generated undirected networks according to the DG-DC model, with two blocks or communities of equal size $n/2$. In order to confound the block model as much as possible, we deliberately designed these networks so that the two blocks have the same average degree. The degree distribution in block 1 is a power law with exponent $\alpha = 1.7$, with an upper bound of 1850, so that the average degree is 20. The degree distribution in block 2 is Poisson, also with mean 20. As described in Appendix D, the upper bound on the power law is larger than any degree actually appearing in the network; it just changes the normalizing constant of the power law, and the MLE for α can still be calculated using (3.17). We assume the algorithm knows that one block has a power law degree distribution and the other is Poisson, but we force it to infer the parameters of these distributions.

As in [33], we use a parameter λ to interpolate linearly between a fully random network with no community structure and a "planted" one where the communities are completely separated. Thus

$$\omega_{rs} = \lambda \omega_{rs}^{\text{planted}} + (1-\lambda)\omega_{rs}^{\text{random}} \tag{3.18}$$

where

$$\omega_{rs}^{\text{random}} = \frac{\kappa_r \kappa_s}{2m}, \; \omega^{\text{planted}} = \begin{pmatrix} \kappa_1 & 0 \\ 0 & \kappa_2 \end{pmatrix}. \tag{3.19}$$

We inferred the community structure with various models. We ran the Kernighan-Lin (KL) heuristic first to find a local optimum [33], and then ran the heat-bath MCMC algorithm with a fixed number of iterations to further refine it if possible. We initialized each run with a random block assignment; to test the stability of the models, we also tried initializing them with the correct block assignment. Since isolated vertices don't participate in the community structure, giving us little or no basis on which we can classify them, we remove them and focus on the giant component. For $\lambda = 1$, where the community structure is purely the "planted" one, we kept two giant components, one in each community.

We measured accuracy by the normalized mutual information (NMI) [19] between the most-likely block assignment found by the model and the correct assignment. To make this more concrete, if there are two blocks of equal size and 95% of the vertices in each block are labeled correctly, the NMI is 0.714. If 90% in each group are labeled correctly, the NMI is 0.531. For groups of unequal size, the NMI is a better measure of accuracy than the fraction of vertices labeled correctly, since one can make this fraction fairly large simply by assigning every vertex to the larger group.

As shown in Fig. 3.2, DG-DC works very well even for small λ. This is because it can classify most of the vertices simply based on their degrees; if d_u is far from 20, for instance, then u is probably in block 1. As λ increases, it uses the connections

between communities as well, giving near-perfect accuracy for $\lambda \geq 0.6$. It does equally well whether its initial assignment is correct or random.

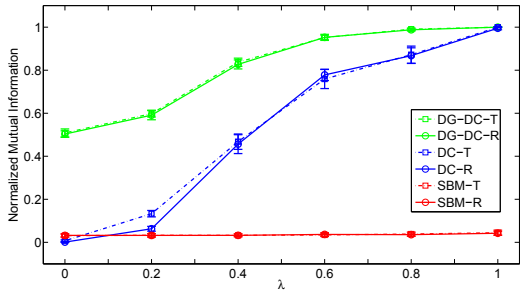


Figure 3.2: Tests on synthetic networks generated by the DG-DC model. Each point is based on 30 randomly generated networks with 2400 nodes. For each network and each model, we choose the best result from 10 independent runs, initialized either with random assignments (the suffix R) or the true block assignment (the suffix T). Each run consisted of the KL heuristic followed by 10^6 MCMC steps. Our degree-generated (DG) block model performs much better on these networks than the degree-corrected (DC) model. The non-degree-corrected (SBM) model doesn't work at all.

The DC model, in contrast, is unable to use the vertex degrees, and has accuracy near zero (i.e., not much better than a random block assignment) for $\lambda \leq 0.2$. Like the SBM [20, 21], it may have a phase transition at a critical value of λ below which the community structure is undetectable. Initializing it with the correct assignment helps somewhat at these values of λ, but even then it settles on an assignment far from the correct one.

The original stochastic block model (SBM), as discussed above, separates vertices with high degrees from vertices with low degrees. Thus it cannot find the correct group structure even for large λ. Our synthetic tests are designed to have a broad degree distribution in block 1, and thus make SBM fail. Note that if the degree distribution in block 1 is a power-law with a larger exponent α, then most of the degrees will be much lower than 20, in which case SBM works reasonably well.

Next, we generated directed networks according to the DG-DDC model. We again have two blocks of equal size, with degree distributions similar to the undirected networks tested above. In block 1, both out- and in-degrees are power-law distributed with $\alpha = 1.7$, with an upper bound of 1850 so that the expected degree is 20. In block 2, both out- and in-degrees are Poisson-distributed with mean 20. To test our oriented and directed models, we interpolate between a random network $\omega_{rs}^{\text{random}} = \kappa_r \kappa_s / 4m$ and a planted network with completely asymmetric connections between the blocks,

$$\omega^{\text{planted}} = \begin{pmatrix} (\kappa_1 - \omega_{12})/2 & \omega_{12} \\ 0 & (\kappa_2 - \omega_{12})/2 \end{pmatrix}, \tag{3.20}$$

where $\omega_{12} \leq \min(\kappa_1, \kappa_2)$. We choose $\omega_{12} = \frac{1}{2}\min(\kappa_1, \kappa_2)$.

As Fig. 3.3 shows, DG-ODC and DG-DDC have very similar performance at the extremes where $\lambda = 0$ and 1. However, DG-ODC works better than DG-DDC for other values of λ, and both of them achieve much better accuracy than the ODC or DDC models. As in Fig. 3.2, the degree-generated models can achieve a high accuracy based simply on the vertex degrees, and as λ grows they leverage this information further to achieve near-perfect accuracy for $\lambda \geq 0.8$.

Among the non-degree-corrected models, ODC performs significantly better than DDC for $\lambda \geq 0.4$. Edges are more likely to point from block 1 to block 2 than vice versa, and ODC can take advantage of this information while DDC cannot. As we will see in the next section, ODC performs well on some real-world networks for precisely this reason.

3.3.2 Experiments on real networks

We studied three word adjacency networks, where vertices are separated into two blocks: adjectives and nouns. The first consists of common words in Dickens' novel

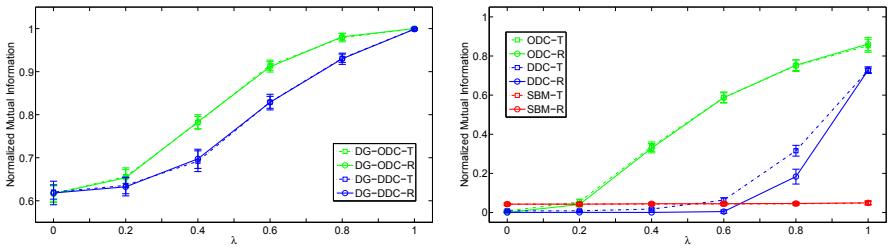


Figure 3.3: Tests on synthetic directed networks with 2400 nodes. Left, DG-ODC and DG-DDC; right, ODC and DDC. The degree-generated models again perform very well even for small λ, since they can use in- and out-degrees to classify the vertices. ODC performs significantly better than DDC for $\lambda \geq 0.4$, since it can use the edge orientations to distinguish the two blocks. The number of networks, runs, and MCMC steps per run are as in Fig. 3.2.

David Copperfield [47]. The other two are built from the Brown corpus, which is a tagged corpus of present-day edited American English across various categories, including news, novels, documents, and many others [26]. The smaller one contains words in the News category (45 archives) that appeared at least 10 times; the larger one contains all the adjectives and nouns in the giant component of the entire corpus.

We considered both the simple version of these networks where $A_{uv} = 1$ if u and v ever occur adjacently in that order, and the multigraph version where $A_{uv} \geq 0$ is the number of adjacent cooccurences. The sizes, block sizes, and number of edges of these networks are shown in Table 3.1. In "News" and "Brown", the block sizes are quite different, with more nouns than adjectives. As discussed above, the NMI is a better measure of accuracy than the fraction of vertices labeled correctly, since we could make the latter fairly large by labeling everything a noun.

In each network, both blocks have heavy-tailed in- and out-degree distributions (Figure 3.4). The connections between them are disassortative and highly asymmetric: since in English adjectives precede nouns more often than they follow them, and more often than adjectives precede adjectives or nouns precede nouns, ω_{12} is roughly 10 times larger than ω_{21}, and ω_{12} is larger than either ω_{11} or ω_{22}. The ω for each

Table 3.1: Basic statistics of the three word adjacency networks. S and M denote the simple and multigraph versions respectively.

Network	#words	#adjective	#noun	#edges (S)	#edges (M)
David	112	57	55	569	1494
News	376	91	285	1389	2411
Brown	23258	6235	17023	66734	88930

Table 3.2: The matrices $\omega_{rs} = m_{rs}/(n_r n_s)$ for the most-likely block assignment according to the stochastic block model.

	David(S)	David(M)	News(S)	News(M)	Brown(S)	Brown(M)
ω_{11}	0.039	0.080	0.010	0.012	9.1e-05	1.1e-04
ω_{12}	0.118	0.358	0.015	0.028	3.4e-04	4.4e-04
ω_{21}	0.018	0.025	0.002	0.003	2.0e-05	2.4e-05
ω_{22}	0.006	0.011	0.010	0.019	8.8e-05	1.2e-04

network corresponding to the correct block assignment (according to the stochastic block model) is shown in Table 3.2.

Performance of oriented and degree-corrected models

Table 3.3 compares the performance of non-degree-generated block models, including SBM, DC, ODC, and DDC. (Under DC, we ignore the edge orientations, and treat the graph as undirected. Note that the resulting network may contain multi-edges even though the directed one doesn't.)

In our experiments, we started with a random initial block assignment, ran the KL heuristic to find a local optimum, and then ran the heat-bath MCMC algorithm. We also tested a naive heuristic NH which simply labels a vertex v as an adjective if $d_v^{\text{out}} > d_v^{\text{in}}$, and a noun if $d_v^{\text{in}} > d_v^{\text{out}}$. If $d_v^{\text{out}} = d_v^{\text{in}}$, NH labels v randomly with equal probabilities.

For "David", DC and ODC work fairly well, and both are better than the naive

Table 3.3: For each model and each network, we pick the block assignment with highest likelihood and compute its NMI with the correct block assignment. Each run consisted of the KL heuristic, starting with a random block assignment, followed by 10^6 MCMC steps. The results for "David" and "News" are based on 100 independent runs; for "Brown", 50 runs are executed. The best NMI for each network is shown in bold.

	David(S)	David(M)	News(S)	News(M)	Brown(S)	Brown(M)
SBM	.423	.051	.006	.018	.001	7e-04
DC	**.566**	**.568**	.084	.083	.020	.015
ODC	.462	.470	.084	.029	**.311**	**.318**
DDC	.128	8e-04	.084	.091	.016	.012
NH	.395	.449	**.215**	**.233**	.309	.314

heuristic NH. Moreover, the mistakes they make are instructive. There are three adjectives with out-degree zero: "full", "glad", and "alone". ODC mislabels these since it expects edges to point away from adjectives, while DC labels them correctly by using the fact that edges are disassortative, tending to cross from one block to the other.

The standard SBM works well on "David(S)" but fails on "David(M)" because the degrees in the multigraph are more skewed than those in the simple one. Finally, DDC performs the worst; by correcting for in- and out-degrees separately, it loses any information that the edge orientations could provide, and even fails to notice the disassortative structure that DC uses. Thus full degree-correction in the directed case can make things worse, even when the degrees in each community are broadly distributed.

For "Brown", all these models fail except ODC, although it does only slightly better than the naive NH. For "News", all these models fail, even ODC. Despite the degree correction, the most-likely block assignment is highly assortative, with high-degree vertices connecting to each other. However, we found that in most runs on "News", ODC used the edge orientations successfully to find the a block assignment

Table 3.4: Results using the naive NH assignment as the initial condition, again followed by 10^6 MCMC steps. This hint now lets ODC outperform the other models on "News". The best NMI for each network is shown in bold.

	David(S)	David(M)	News(S)	News(M)	Brown(S)	Brown(M)
SBM	.423	.051	.006	.021	.001	7e-04
DC	**.566**	**.568**	.084	.015	.160	.155
ODC	.462	.470	**.247**	**.270**	**.311**	**.318**
DDC	.015	.060	.084	.005	.005	.070
NH	.395	.449	.215	.233	.309	.314

close to the correct one; it found the assortative structure only occasionally. This suggests that, even though the "wrong" structure has a higher likelihood, we can do much better if we know what kind of community structure to look for; in this case, disassortative and directed.

To test this hypothesis, we tried giving the models a hint about the community structure by using NH to determine the initial block assignment. We then performed the KL heuristic and the MCMC algorithm as before. As Table 3.4 shows, this hint improves ODC's performance on "News" significantly; it is able to take the initial naive classification, based solely on degrees, and refine it using the network's structure. Note that this more accurate assignment actually has lower likelihood than the one found in Table 3.3 using a random initial condition—so NH helps the model stay in a more accurate, but less likely, local optimum. Starting with NH improves DC's performance on "Brown" somewhat, but DC still ends up with an assignment less accurate than the naive one.

Performance of degree-generated models

In this section, we measure the performance of degree-generated models on the Brown network, and compare them to their non-degree-generated counterparts. According to Figure 3.4, the in- and out-degree distributions in each block have heavy tails

close to a power-law. Moreover, the out-degrees of the adjectives have a heavier tail than those of the nouns, and vice versa for the in-degrees. This is exactly the kind of difference in the degree distributions between communities that our DG block models are designed to take advantage of.

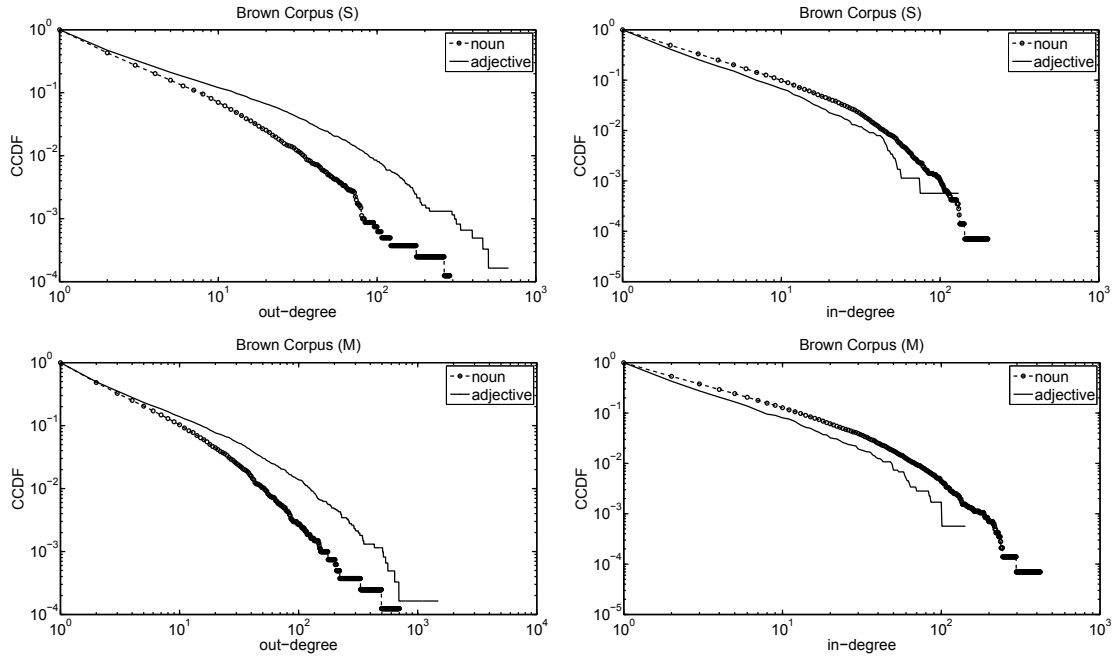


Figure 3.4: Degree distributions in the Brown network.

Table 3.5: MLEs for the degree generation parameters in the Brown network, given the correct assignment.

	Brown(S)				Brown(M)			
block	$\hat{\alpha}_{\text{in}}$	$\hat{\alpha}_{\text{out}}$	$\hat{\beta}_{\text{in}}$	$\hat{\beta}_{\text{out}}$	$\hat{\alpha}_{\text{in}}$	$\hat{\alpha}_{\text{out}}$	$\hat{\beta}_{\text{in}}$	$\hat{\beta}_{\text{out}}$
adjective	2.329	2.629	0.161	0.527	2.136	2.326	0.161	0.527
noun	2.721	2.248	0.716	0.021	2.576	2.134	0.716	0.021

Setting $S_{\min} = 1$, we can estimate the parameters α and β for these distributions as discussed in Section 3.2. We show the most likely values of these parameters, given the correct assignment, in Table 3.5.

Table 3.6: Performance of degree-generated models on the Brown corpus. KL indicates that we applied the KL heuristic after 10^6 MCMC steps. DG indicates degree generation. Each number gives the NMI for the most-likely assignment found in 50 independent runs. The best model is DG-ODC. Moreover, degree generation helps ODC converge, providing much of the benefit of the KL heuristic while avoiding its long running time (see bold numbers).

		Brown(S)			Brown(M)		
		DC	ODC	DDC	DC	ODC	DDC
–	–	.010	.188	.008	.007	.203	.011
KL	–	.020	.311	.016	.015	.318	.012
–	DG	.267	.302	.213	.278	.310	.149
KL	DG	.271	**.312**	.225	.284	**.320**	.195

As Table 3.6 shows, degree generation improves DC and DDC significantly, letting them find a good assignment as opposed to one with NMI near zero. For ODC, the slight performance improvement makes DG-ODC the best model overall. We compare performance starting with the KL heuristic to performance using MCMC alone. We see that degree generation gives ODC almost as much benefit as the KL heuristic does. In other words, it speeds up the MCMC optimization process, letting ODC find a good assignment without the initial help of the computationally expensive KL heuristic.

3.4 Summary

Based on the degree-corrected (DC) block model for undirected networks introduced in [33], we developed two block models with degree-correction for directed networks, which are the directed degree-corrected (DDC) and the oriented degree-corrected (ODC) block models. When applying these models on directed networks, DC totally ignores the direction of the edges and generates the undirected version of the given network. DC prefers community structures where the degrees of the nodes are highly

skewed in each community. DDC generates the directed network completely, thus both the in- and out-degrees are corrected. In other words, DDC prefers community structures where in each community both the in- and out-degrees of the nodes are highly skewed. Thus DDC is unable to use the edge orientation informaiton when the community structure presents disassortative mixing and the inter-community connections are highly directed. ODC works perfectly in such a case by generating the undirected network first using the DC model and then generating the edge orientations. ODC prefers disassortative mixing and highly directed inter-community connections.

Non-degree-corrected and degree-corrected block models represent two extremes: one prefers homogeneous degree distributions in each community and the other prefers inhomogeneous degree distributions. Degree generation (DG) fills the gap between them. DG generates expected degrees of the nodes first and then generates the edges using degree-corrected block models, which can be DC, DDC or ODC. DG works jointly with degree-corrected block models so that the joint model is able to use the prior knowledge of the degree distributions in each community, which can be homogeneous, inhomogeneous or any specified distributions.

The lesson we learned here about generative models and statistical inference is that when inferring the structure of a network, you can only use the information that you try to generate. ODC generates the edge orientations so that the model can use this information for community detection; DG generates the degree distributions, thus it's able to use the information of the degree distributions. In the next chapter, we will see how we can use node attributes to help us infer the communities. Guided by the same philosophy, we developed models which are able to use information beyond the network topology by simply generating rich attributes of the nodes.

Chapter 4

Scalable text and link analysis

The work described in this chapter is published in [60] under the supervision of Prof. Cristopher Moore. I also collaborated with Xiaoran Yan and Lise Getoor. Xiaoran helped with the writing. Lise provided some useful tools for data processing. Lise's expertise on machine learning and topic modeling helped the development of the paper. I collaborated with Sergi Valverde on the project of the patent citation networks. Sergi is an expert on the evolution of technology, and he provided us the patent network data.

Many modern data sets contain both rich information about each object, and pairwise relationships between them, forming networks where each object is a node and links represent the relationships. In document networks, for example, each node is a document containing a sequence of words, and the links between nodes are citations or hyperlinks. Both the content of the documents and the topology of the links between them are meaningful.

Over the past few years, two disparate communities have been approaching these data sets from different points of view. In the data mining community, the goal has

been to augment traditional approaches to learning and data mining by including relations between objects [27, 37, 57]: for instance, using the links between documents to help us label them by topic. In the network community, including its subset in statistical physics, the goal has been to augment traditional community structure algorithms such as the stochastic block model [23, 31, 51] by taking node attributes into account, for instance, to use the content of documents, rather than just the topological links between them, to help us understand their community structure.

In the original stochastic block model, each node has a discrete label, assigning it to one of k communities. These labels, and the $k \times k$ matrix of probabilities with which a given pair of nodes with a given pair of labels have a link between them, can be inferred using Monte Carlo algorithms (e.g. [42]) or, more efficiently, with belief propagation [21, 20] or pseudolikelihood approaches [14]. However, in real networks communities often overlap, and a given node can belong to multiple communities. This led to the *mixed-membership* block model [2], where the goal is to infer, for each node v, a distribution or mixture of labels θ_v describing to what extent it belongs to each community. If we assume that links are assortative, i.e., that nodes are more likely to link to others in the same community, then the probability of a link between two nodes v and v' depends on some measure of similarity (say, the inner product) of θ_v and $\theta_{v'}$.

These mixed-membership block models fit nicely with classic ideas in topic modeling. In models such as Probabilistic Latent Semantic Analysis (PLSA) [30] and Latent Dirichlet Allocation (LDA) [11], each document d has a mixture θ_d of topics. Each topic corresponds in turn to a probability distribution over words, and each word in d is generated independently from the resulting mixture of distributions. If we think of θ_d as both the mixture of topics for generating words and the mixture of communities for generating links, then we can infer $\{\theta_d\}$ jointly from the documents' content and the presence or absence of links between them.

There are many possible such models, and we are far from the first to think along these lines. Our innovation is to take as our starting point a particular mixed-membership block model recently developed in the physics community [7], which we call the BKN model. It differs from the *mixed-membership stochastic block model* (MMSB) of [2] in several ways:

1. The BKN model treats the community membership mixtures θ_d directly as parameters to be inferred. In contrast, MMSB treats θ_d as hidden variables generated by a Dirichlet distribution, and infers the hyperparameters of that distribution. The situation between PLSA and LDA is similar; PLSA infers the topic mixtures θ_d, while LDA generates them from a Dirichlet distribution.

2. The MMSB model generates each link according to a Bernoulli distribution, with an extra parameter for sparsity. Instead, BKN treats the links as a random multigraph, where the number of links $A_{dd'}$ between each pair of nodes is Poisson-distributed. As a result, the derivatives of the log-likelihood with respect to θ_d and the other parameters are particularly simple.

These two factors make it possible to fit the BKN model using an efficient and exact expectation-maximization (EM) algorithm, making its inference highly scalable. The BKN model has another advantage as well:

3. The BKN model is *degree-corrected*, in that it takes the observed degrees of the nodes into account when computing the expected number of edges between them. Thus it recognizes that two documents that have very different degrees might in fact have the same mix of topics; one may simply be more popular than the other.

In our work, we use a slight variant of the BKN model to generate the links, and we use PLSA to generate the text. We present an EM algorithm for inferring the

topic mixtures and other parameters. (While we do not impose a Dirichlet prior on the topic mixtures, it is easy to add a corresponding term to the update equations.) Our algorithm is scalable in the sense that each iteration takes $O(K(N + M + R))$ time for networks with K topics, N documents, and M links, where R is the sum over documents of the number of distinct words appearing in each one. In practice, our EM algorithm converges within a small number of iterations, making the total running time linear in the size of the corpus.

Our model can be used for a variety of learning and generalization tasks, including document classification or link prediction. For document classification, we can obtain hard labels for each document by taking its most-likely topic with respect to θ_d, and optionally improve these labels further with local search. For link prediction, we train the model using a subset of the links, and then ask it to rank the remaining pairs of documents according to the probability of a link between them. For each task we determine the optimal relative weight of the content vs. the link information.

We performed experiments on three real-world data sets, with thousands of documents and millions of words. The experimental results illustrated in Section 4.3 show that our algorithm is more accurate, and considerably faster, than previous techniques for both document classification and link prediction.

The rest of the chapter is organized as follows. Section 4.1 describes our generative model, and compares it with related models in the literature. Section 4.2 gives our EM algorithm and analyzes its running time.

4.1 Our model and previous work

In this section, we give our proposed model, which we call the *Poisson mixed-topic link model* (PMTLM) and its degree-corrected variant PMTLM-DC.

4.1.1 The generative model

Consider a network of N documents. Each document d has a fixed length L_d, and consists of a string of words $w_{d\ell}$ for $1 \le \ell \le L_d$, where $1 \le w_{d\ell} \le W$ where W is the number of distinct words. In addition, each pair of documents d, d' has an integer number of links connecting them, giving an adjacency matrix $A_{dd'}$. There are K topics, which play the dual role of the overlapping communities in the network.

Our model generates both the content $\{w_{d\ell}\}$ and the links $\{A_{dd'}\}$ as follows. We generate the content using the PLSA model [30]. Each topic z is associated with a probability distribution β_z over words, and each document has a probability distribution θ_d over topics. For each document $1 \le d \le N$ and each $1 \le \ell \le L_d$, we independently choose a topic $z = z_{d\ell} \sim \text{Multi}(\theta_d)$, and choose the word $w_{d\ell} \sim \text{Multi}(\beta_z)$. Thus the total probability that $w_{d\ell}$ is a given word w is

$$\Pr[w_{d\ell} = w] = \sum_{z=1}^{K} \theta_{dz}\beta_{zw} \,. \tag{4.1}$$

We assume that the number of topics K is fixed. The distributions β_z and θ_d are parameters to be inferred.

We generate the links using a version of the Ball-Karrer-Newman (BKN) model [7]. Each topic z is associated with a link density η_z. For each pair of documents d, d' and each topic z, we independently generate a number of links which is Poisson-distributed with mean $\theta_{dz}\theta_{d'z}\eta_z$. Since the sum of independent Poisson variables is Poisson, the total number of links between d and d' is distributed as

$$A_{dd'} \sim \text{Poi}\left(\sum_z \theta_{dz}\theta_{d'z}\eta_z\right) \,. \tag{4.2}$$

Since $A_{dd'}$ can exceed 1, this gives a random multigraph. In the data sets we study below, $A_{dd'}$ is 1 or 0 depending on whether d cites d', giving a simple graph. On the other hand, in the sparse case the event that $A_{dd'} > 1$ has low probability in

our model. Moreover, the fact that $A_{dd'}$ is Poisson-distributed rather than Bernoulli makes the derivatives of the likelihood with respect to the parameters θ_{dz} and η_z very simple, allowing us to write down an efficient EM algorithm for inferring them.

This version of the model assumes that links are assortative, i.e., that links between documents only form to the extent that they belong to the same topic. One can easily generalize the model to include disassortative links as well, replacing η_z with a matrix $\eta_{zz'}$ that allows documents with distinct topics z, z' to link [7].

We also consider *degree-corrected* versions of this model, where in addition to its topic mixture θ_d, each document has a propensity S_d of forming links. In that case,

$$A_{dd'} \sim \text{Poi}\left(S_d S_{d'} \sum_z \theta_{dz}\theta_{d'z}\eta_z\right) . \tag{4.3}$$

We call this variant the *Poisson Mixed-Topic Link Model with Degree Correction* (PMTLM-DC).

4.1.2 Prior work on content–link models

Most models for document networks generate content using either PLSA [30], as we do, or LDA [11]. The distinction is that PLSA treats the document mixtures θ_d as parameters, while in LDA they are hidden variables, integrated over a Dirichlet distribution. As we show in Section 4.2, our approach gives a simple, exact EM algorithm, avoiding the need for sampling or variational methods. While we do not impose a Dirichlet prior on θ_d in this paper, it is easy to add a corresponding term to the update equations for the EM algorithm, with no loss of efficiency.

There are a variety of methods in the literature to generate links between documents. PHITS-PLSA [18], LINK-LDA [22] and LINK-PLSA-LDA [44] use the PHITS [17] model for link generation. PHITS treats each document as an additional term in the vocabulary, so two documents are similar if they link to the same documents. This is

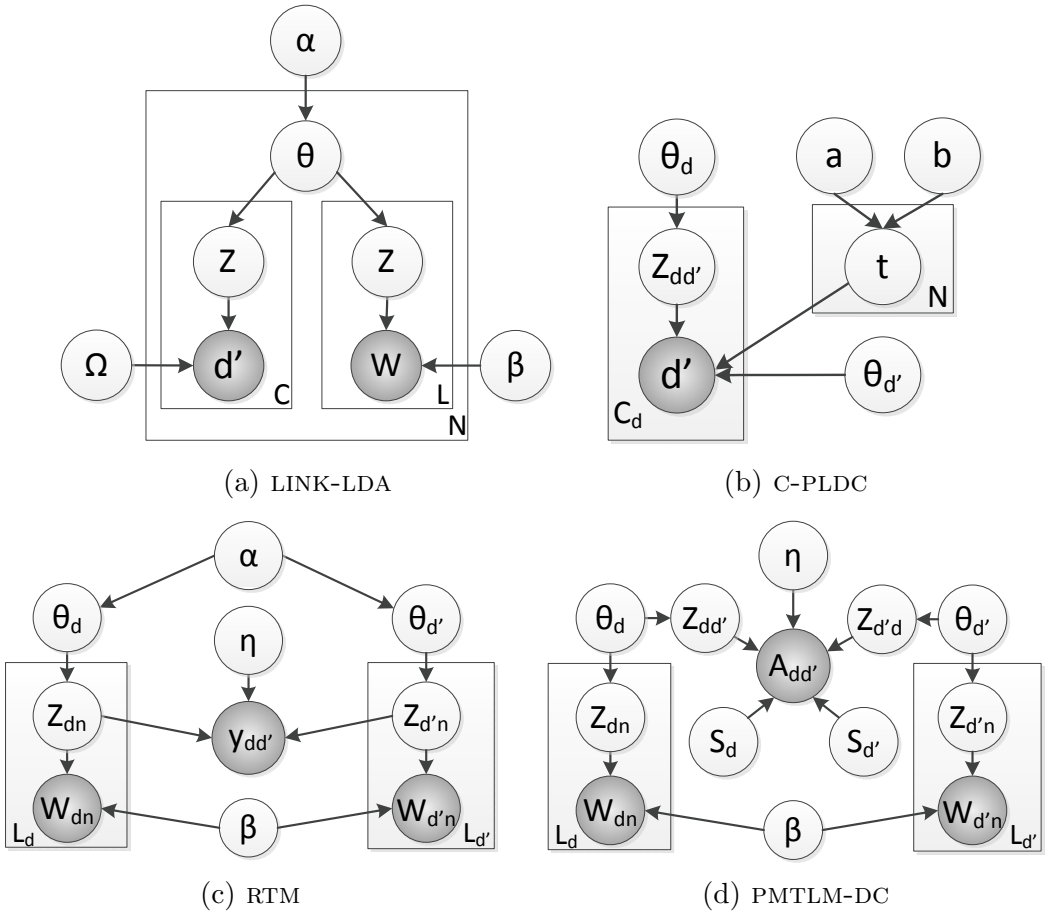


Figure 4.1: Graphical models for link generation.

analogous to a mixture model for networks studied in [48]. In contrast, block models like ours treat documents as similar if they link to *similar* documents, as opposed to literally the same ones.

The PAIRWISE LINK-LDA model [44], like ours, generates the links with a mixed-topic block model, although as in MMSB [2] and LDA [11] it treats the θ_d as hidden variables integrated over a Dirichlet prior. They fit their model with a variational method that requires N^2 parameters, making it less scalable than our approach.

In the C-PLDC model [56], the link probability from d to d' is determined by their

topic mixtures $\theta_d, \theta_{d'}$ and the popularity $t_{d'}$ of d', which is drawn from a Gamma distribution with hyperparameters a and b. Thus $t_{d'}$ plays a role similar to the degree-correcting parameter $S_{d'}$ in our model, although we correct for the degree of d as well. However, C-PLDC does not generate the content, but takes it as given.

The Relational Topic Model (RTM) [12, 13] assumes that the link probability between d and d' depends on the topics of the words appearing in their text. In contrast, our model uses the underlying topic mixtures θ_d to generate both the content and the links. Like our model, RTM defines the similarity of two topics as a weighted inner product of their topic mixtures: however, in RTM the probability of a link is a nonlinear function of this similarity, which can be logistic, exponential or normal, of this similarity.

Although it deals with a slightly different kind of dataset, our model is closest in spirit to the Latent Topic Hypertext Model (LTHM) [29]. This is a generative model for hypertext networks, where each link from d to d' is associated with a specific word w in d. If we sum over all words in d, the total number of links $A_{dd'}$ from d to d' that LTHM would generate follows a binomial distribution

$$A_{dd'} \sim \mathrm{Bin}\left(L_d, \lambda_{d'} \sum_z \theta_{dz}\theta_{d'z}\right) , \tag{4.4}$$

where $\lambda_{d'}$ is, in our terms, a degree-correction parameter. When L_d is large this becomes a Poisson distribution with mean $L_d\lambda_{d'}\sum_z \theta_{dz}\theta_{d'z}$. Our model differs from this in two ways: our parameters η_z give a link density associated with each topic z, and our degree correction S_d does not assume that the number of links from d is proportional to its length.

We briefly mention several other approaches. The authors of [27] extend the probabilistic relational model (PRM) framework and proposed a unified generative model for both content and links in a relational structure. In [38], the authors proposed a link-based model that describes both node attributes and links. The

HTM model [52] treats links as fixed rather than generating them, and only generates the text. Finally, the LMMG model [34] treats the appearance or absence of a word as a binary attribute of each document, and uses a logistic or exponential function of these attributes to determine the link probabilities.

In Section 4.3 below, we compare our model to PHITS-PLSA, LINK-LDA, C-PLDC, and RTM. Graphical models for the link generation components of these models, and ours, are shown in Figure 4.1.

4.2 A scalable EM algorithm

Here we describe an efficient Expectation-Maximization algorithm to find the MLEs of the parameters in our model. Each update takes $O(K(N + M + R))$ time for a document network with K topics, N documents, and M links, where R is the sum over the documents of the number of distinct words in each one. Thus the running time per iteration is linear in the size of the corpus.

For simplicity we describe the algorithm for the simpler version of our model, PMTLM. The algorithm for the degree-corrected version, PMTLM-DC, is similar (see Appendix F).

4.2.1 The likelihood

Let C_{dw} denote the number of times a word w appears in document d. From (4.1), the log-likelihood of d's content is

$$\begin{aligned}\mathcal{L}_d^{\text{content}} &= \log P(w_{d1}, \ldots, w_{dL_d} \,|\, \theta_d, \beta) \\ &= \sum_{w=1}^{W} C_{dw} \log \left(\sum_{z=1}^{K} \theta_{dz} \beta_{zw} \right) . \end{aligned} \tag{4.5}$$

Similarly, from (4.2), the log-likelihood for the links $A_{dd'}$ is

$$\begin{aligned}\mathcal{L}^{\text{links}} &= \log P(A \,|\, \theta, \eta) \\ &= \frac{1}{2} \sum_{dd'} A_{dd'} \log \left(\sum_{z} \theta_{dz} \theta_{d'z} \eta_z \right) - \frac{1}{2} \sum_{dd'} \sum_{z} \theta_{dz} \theta_{d'z} \eta_z \, . \end{aligned} \tag{4.6}$$

We ignore the constant term $-\sum_{dd'} \log A_{dd'}!$ from the denominator of the Poisson distribution, since it has no bearing on the parameters.

4.2.2 Balancing content and links

While we can use the total likelihood $\sum_d \mathcal{L}_d^{\text{content}} + \mathcal{L}^{\text{links}}$ directly, in practice we can improve our performance significantly by better balancing the information in the content vs. that in the links. In particular, the log-likelihood $\mathcal{L}_d^{\text{content}}$ of each document is proportional to its length, while its contribution to $\mathcal{L}^{\text{links}}$ is proportional to its degree. Since a typical document has many more words than links, $\mathcal{L}^{\text{content}}$ tends to be much larger than $\mathcal{L}^{\text{links}}$.

Following [30], we can provide this balance in two ways. One is to normalize $\mathcal{L}^{\text{content}}$ by the length L_d, and another is to add a parameter α that reweights the relative contributions of the two terms $\mathcal{L}^{\text{content}}$ and $\mathcal{L}^{\text{links}}$. We then maximize

$$\mathcal{L} = \alpha \sum_{d} \frac{1}{L_d} \mathcal{L}_d^{\text{content}} + (1 - \alpha) \mathcal{L}^{\text{links}} \, . \tag{4.7}$$

Varying α from 0 to 1 lets us interpolate between two extremes: studying the document network purely in terms of its topology, or purely in terms of the documents'

content. Indeed, we will see in Section 4.3 that the optimal value of α depends on which task we are performing: closer to 0 for link prediction, and closer to 1 for topic classification.

4.2.3 Update equations and running time

We maximize $\mathcal{L}$ as a function of $\{\theta, \beta, \eta\}$ using an EM algorithm, very similar to the one introduced by [7] for overlapping community detection. We start with a standard trick to change the log of a sum into a sum of logs, writing

$$\begin{aligned}
\mathcal{L}_d^{\text{content}} &\geq \sum_{w=1}^{W} C_{dw} \sum_{z=1}^{K} h_{dw}(z) \log \frac{\theta_{dz}\beta_{zw}}{h_{dw}(z)} \\
\mathcal{L}^{\text{links}} &\geq \frac{1}{2} \sum_{dd'} \sum_{z=1}^{K} A_{dd'} q_{dd'}(z) \log \frac{\theta_{dz}\theta_{d'z}\eta_z}{q_{dd'}(z)} - \frac{1}{2} \sum_{dd'} \sum_{z=1}^{K} \theta_{dz}\theta_{d'z}\eta_z \,. \qquad (4.8)
\end{aligned}$$

Here $h_{dw}(z)$ is the probability that a given appearance of w in d is due to topic z, and $q_{dd'}(z)$ is the probability that a given link from d and d' is due to topic z. This lower bound holds with equality when

$$h_{dw}(z) = \frac{\theta_{dz}\beta_{zw}}{\sum_{z'} \theta_{dz'}\beta_{z'w}} \,,\; q_{dd'}(z) = \frac{\theta_{dz}\theta_{d'z}\eta_z}{\sum_{z'} \theta_{dz'}\theta_{d'z'}\eta_{z'}} \,, \qquad (4.9)$$

giving us the E step of the algorithm.

For the M step, we derive update equations for the parameters $\{\theta, \beta, \eta\}$. By taking derivatives of the log-likelihood (4.7) (see the online version for details) we obtain

$$\eta_z = \frac{\sum_{dd'} A_{dd'} q_{dd'}(z)}{\left(\sum_d \theta_{dz}\right)^2} \qquad (4.10)$$

$$\beta_{zw} = \frac{\sum_d (1/L_d)\, C_{dw} h_{dw}(z)}{\sum_d (1/L_d) \sum_{w'} C_{dw'} h_{dw'}(z)} \qquad (4.11)$$

$$\theta_{dz} = \frac{(\alpha/L_d) \sum_w C_{dw} h_{dw}(z) + (1-\alpha) \sum_{d'} A_{dd'} q_{dd'}(z)}{\alpha + (1-\alpha)\kappa_d} \,. \qquad (4.12)$$

Here $\kappa_d = \sum_{d'} A_{dd'}$ is the degree of document d.

To analyze the running time, let R_d denote the number of distinct words in document d, and let $R = \sum_d R_d$. Then only KR of the parameters $h_{dw}(z)$ are nonzero. Similarly, $q_{dd'}(z)$ only appears if $A_{dd'} \neq 0$, so in a network with M links only KM of the $q_{dd'}(z)$ are nonzero. The total number of nonzero terms appearing in (4.9)–(4.12), and hence the running time of the E and M steps, is thus $O(K(N + M + R))$.

As in [7], we can speed up the algorithm if θ is sparse, i.e. if many documents belong to fewer than K topics, so that many of the θ_{dz} are zero. According to (4.9), if $\theta_{dz} = 0$ then $h_{d\ell}(z) = q_{dd'}(z) = 0$, in which case (4.12) implies that $\theta_{dz} = 0$ for all future iterations. If we choose a threshold below which θ_{dz} is effectively zero, then as θ becomes sparser we can maintain just those $h_{d\ell}(z)$ and $q_{dd'}(z)$ where $\theta_{dz} \neq 0$. This in turn simplifies the updates for η and β in (4.10) and (4.11).

We note that the simplicity of our update equations comes from the fact that the $A_{dd'}$ is Poisson, and that its mean is a multilinear function of the parameters. Models where $A_{dd'}$ is Bernoulli-distributed with a more complicated link probability, such as a logistic function, have more complicated derivatives of the likelihood, and therefore more complicated update equations.

Note also that this EM algorithm is exact, in the sense that the maximum-likelihood estimators $\{\widehat{\theta}, \widehat{\beta}, \widehat{\eta}\}$ are fixed points of the update equations. This is because the E step (4.9) is exact, since the conditional distribution of topics associated with each word occurrence and each link is a product distribution, which we can describe exactly with h_{dw} and $q_{dd'}$. (There are typically multiple fixed points, so in practice we run our algorithm with many different initial conditions, and take the fixed point with the highest likelihood.)

This exactness is due to the fact that the topic mixtures θ_d are parameters to

be inferred. In models such as LDA and MMSB where θ_d is a hidden variable integrated over a Dirichlet prior, the topics associated with each word and link have a complicated joint distribution that can only be approximated using sampling or variational methods. (To be fair, recent advances such as stochastic optimization based on network subsampling [28] have shown that approximate inference in these models can be carried out quite efficiently.)

On the other hand, in the context of finding communities in networks, models with Dirichlet priors have been observed to generalize more successfully than Poisson models such as BKN [28]. Happily, we can impose a Dirichlet prior on θ_d with no loss of efficiency, simply by including pseudocounts in the update equations—in essence adding additional words and links that are known to come from each topic. This lets us obtain a maximum a posteriori (MAP) estimate of an LDA-like model. We leave this as a direction for future work.

4.2.4 Discrete labels and local search

Our model, like PLSA and the BKN model, lets us infer a soft classification—a mixture of topic labels or community memberships for each document. However, we often want to infer categorical labels, where each document d is assigned to a single topic $1 \leq z_d \leq K$. A natural way to do this is to let z_d be the most-likely label in the inferred mixture, $\hat{z}_d = \text{argmax}_z \, \theta_{dz}$. This is equivalent to rounding θ_d to a delta function, $\theta_{dz} = 1$ for $z = \hat{z}_d$ and 0 for $z \neq \hat{z}_d$.

If we wish, we can improve these discrete labels further using local search. If each

document has just a single topic, the log-likelihood of our model is

$$\mathcal{L}_d^{\text{content}} = \sum_{w=1}^{W} C_{dw} \log \beta_{z_d w} \tag{4.13}$$

$$\mathcal{L}^{\text{links}} = \frac{1}{2} \sum_{dd'} A_{dd'} \log \eta_{z_d z_{d'}} \,. \tag{4.14}$$

Note that here η is a matrix, with off-diagonal entries that allow documents with different topics $z_d, z_{d'}$to be linked. Otherwise, these discrete labels would cause the network to split into K separate components.

Let n_z denote the number of documents of topic z, let $L_z = \sum_{d:z_d=z} L_d$ be their total length, and let $C_{zw} = \sum_{d:z_d=z} C_{dw}$ be the total number of times w appears in them. Let $m_{zz'}$ denote the total number of links between documents of topics z and z', counting each link twice if $z = z'$. Then the MLEs for β and η are

$$\hat{\beta}_{zw} = \frac{C_{zw}}{L_z} \,,\; \hat{\eta}_{zz'} = \frac{m_{zz'}}{n_z n_{z'}} \,. \tag{4.15}$$

Applying these MLEs in (4.13) and (4.14) gives us a point estimate of the likelihood of a discrete topic assignment z_d, which we can normalize or reweight as discussed in Section 4.2.2 if we like. We can then maximize this likelihood using local search: for instance, using the Kernighan-Lin heuristic as in [33] or a Monte Carlo algorithm to find a local maximum of the likelihood in the vicinity of $\hat{z}$. Each step of these algorithms changes the label of a single document d, so we can update the values of n_z, L_z, C_{zw}, and $m_{zz'}$ and compute the new likelihood in $O(K + R_d + \kappa_d)$ time. In our experiments we used the KL heuristic, and found that for some data sets it noticeably improved the accuracy of our algorithm for the document classification task.

4.3 Experimental results

In this section we present empirical results on our model and our algorithm for unsupervised document classification and link prediction. We compare its accuracy and running time with those of several other methods, testing it on three real-world document citation networks.

4.3.1 Data sets

The top portion of Table 4.1 lists the basic statistics for three real-world corpora [50]: Cora, Citeseer, and PubMed[1]. Cora and Citeseer contain papers in machine learning, with $K = 7$ topics for Cora and $K = 6$ for Citeseer. PubMed consists of medical research papers on $K = 3$ topics, namely three types of diabetes. All three corpora have ground-truth topic labels provided by human curators.

The data sets for the three corpora are slightly different. PubMed contains the number of times C_{dw} each word appeared in each document, while Cora and Citeseer record whether or not a word occurred at least once in the document. For Cora and Citeseer, we treat C_{dw} as 0 or 1.

4.3.2 Models and implementations

We compare the Poisson Mixed-Topic Link Model (PMTLM) and its degree-corrected variant, denoted PMTLM-DC, with PHITS-PLSA, LINK-LDA, C-PLDC, and RTM (see Section 4.1.2). We used our own implementation of both PHITS-PLSA and RTM. For RTM, we implemented the variational EM algorithm given in [13]. The implementation is

[1]These data sets are available for download at `http://www.cs.umd.edu/projects/linqs/projects/lbc/`

based on the LDA code available from the authors[2]. We also tried the code provided by J. Chang[3], which uses a Monte Carlo algorithm for the E step, but we found the variational algorithm works better on our data sets. While RTM includes a variety of link probability functions, we only used the sigmoid function. We also assume a symmetric Dirichlet prior. The results for LINK-LDA and C-PLDC are taken from [56].

Each E and M step of the variational algorithm for RTM performs multiple iterations until they converge on estimates for the posterior and the parameters [13]. This is quite different from our EM algorithm: since our E step is exact, we update the parameters only once in each iteration. Our convergence condition for the E step and for the entire EM algorithm are that the fractional increase of the log-likelihood between iterations is less than 10^{-6}; we performed a maximum of 50 iterations in each E step and a maximum of 500 EM iterations for the entire algorithm. To optimize the η parameters (see the graphical model in Section 4.1.2) RTM uses a tunable regularization parameter ρ, which can be thought of as the number of observed non-links. We tried various settings for ρ, namely $0.1M, 0.2M, 0.5M, M, 2M, 5M$ and $10M$ where M is the number of observed links, and tuned ρ separately for each data set and each task. We used gradient descent to optimize the η parameters in each M step.

As described in Section 4.2.2, for PMTLM, PMTLM-DC and PHITS-PLSA we vary the relative weight α of the likelihood of the content vs. the links, tuning α to its best possible value for each data set and each task. For the PubMed data set, we also normalized the content likelihood by the length of the documents.

[2]See `http://www.cs.princeton.edu/~blei/lda-c/`
[3]See `http://www.cs.princeton.edu/~blei/lda/`

4.3.3 Document classification

Experimental setting

For PMTLM, PMTLM-DC and PHITS-PLSA, we performed 500 independent runs of the EM algorithm, each with random initial values of the parameters and topic mixtures. For each run we iterated the EM algorithm up to 5000 times; we found that it typically converges in fewer iterations, with the criterion that the fractional increase of the log-likelihood for two successive iterations is less than 10^{-7}. Figure 4.2 shows that the log-likelihood as a function of the number of iterations are quite similar for all three data sets, even though these corpora have very different sizes. This indicates that even for large data sets, our algorithm converges within a small number of iterations, making its total running time linear in the size of the corpus.

For PMTLM and PMTLM-DC, we obtain discrete topic labels by running our EM algorithm and rounding the topic mixtures as described in Section 4.2.4. We also tested improving these labels with local search, using the Kernighan-Lin heuristic to change the label of one document at a time until we reach a local optimum of the likelihood. More precisely, of those 500 runs, we took the T best fixed points of the EM algorithm (i.e., with the highest likelihood) and attempted to improve them further with the KL heuristic. We used $T = 50$ for Cora and Citeseer and $T = 5$ for PubMed.

For RTM, in each E step, we initialize the variational parameters randomly, and in each M step we initialize the hyperparameters randomly. We execute 500 independent runs for each setting of the tunable parameter ρ.

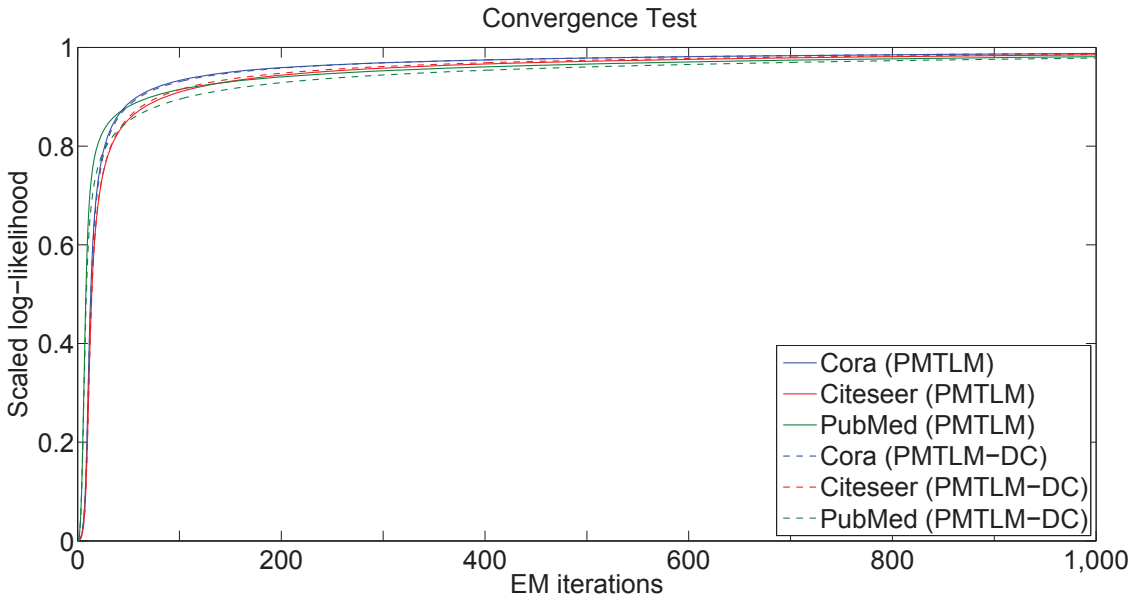


Figure 4.2: The average log-likelihood of the PMTLM and PMTLM-DC models as a function of the number of EM iterations, normalized so that 0 and 1 are the initial and final log-likelihood for 5000 EM iterations. Each points is the average over 100 independent runs. In both models and all three data sets, we approach 1 after just 1000 iterations, showing that the convergence time is roughly constant as a function of the size of the corpus.

Metrics

For each algorithm, we used several measures of the accuracy of the inferred labels as compared to the human-curated ones. The Normalized Mutual Information (NMI) between two labelings C_1 and C_2 is defined as

$$\mathrm{NMI}(C_1, C_2) = \frac{\mathrm{MI}(C_1, C_2)}{\max(\mathrm{H}(C_1), \mathrm{H}(C_2))}\,. \tag{4.16}$$

Here $\mathrm{MI}(C_1, C_2)$ is the mutual information between C_1 and C_2, and $\mathrm{H}(C_1)$ and $\mathrm{H}(C_2)$ are the entropies of C_1 and C_2 respectively. Thus the NMI is a measure of how much information the inferred labels give us about the true ones. We also used the Pairwise F-measure (PWF) [8] and the Variation of Information (VI) [40] (which we wish to minimize).

		Cora	Citeseer	PubMed
Statistics	K	7	6	3
	N	2,708	3,312	19,717
	M	5,429	4,608	44,335
	W	1,433	3,703	4,209
	R	49,216	105,165	1,333,397
Time (sec)	EM (PLSA)	28	61	362
	EM (PHITS-PLSA)	40	67	445
	EM (PMTLM)	33	64	419
	EM (PMTLM-DC)	36	64	402
	EM (RTM)	992	597	2,194
	KL (PMTLM)	375	618	13,723
	KL (PMTLM-DC)	421	565	13,014

Table 4.1: The statistics of the three data sets, and the mean running time, for the EM algorithms in our model PMTLM, its degree-corrected variant PMTLM-DC, and PLSA, PHITS-PLSA, and RTM. Each corpus has K topics, N documents, M links, a vocabulary of size W, and a total size R. Running times for our algorithm, PLSA, and PHITS-PLSA are given for one run of 5000 EM iterations. Running times for RTM consist of up to 500 EM iterations, or until the convergence criteria are reached. Our EM algorithm is highly scalable, with a running time that grows linearly with the size of the corpus. In particular, it is much faster than the variational algorithm for RTM. Improving discrete labels with the Kernighan-Lin heuristic (KL) increases our algorithm's running time, but improves its accuracy for document classification in Cora and Citeseer.

	Cora			Citeseer			PubMed		
Algorithm	NMI	VI	PWF	NMI	VI	PWF	NMI	VI	PWF
PHITS-PLSA	0.382 (.4)	2.285 (.4)	0.447 (.3)	0.366 (.5)	2.226 (.5)	0.480 (.5)	0.233 (1.0)	1.633 (1.0)	0.486 (1.0)
LINK-LDA	0.359†	—	0.397†	0.192†	—	0.305†	—	—	—
C-PLDC	0.489†	—	0.464†	0.276†	—	0.361†	—	—	—
RTM	0.349	2.306	0.422	0.369	2.209	0.480	0.228	1.646	0.482
PMTLM	0.467 (.4)	1.957 (.4)	0.509 (.3)	0.399 (.4)	2.106 (.4)	0.509 (.3)	0.232 (.9)	1.639 (1.0)	0.486 (.9)
PMTLM (KL)	**0.514** (.4)	**1.778** (.4)	**0.525** (.4)	**0.414** (.6)	**2.057** (.6)	0.518 (.5)	0.233 (.9)	1.642 (.9)	0.488 (.9)
PMTLM-DC	0.474 (.3)	1.930 (.3)	0.498 (.3)	0.402 (.3)	2.096 (.3)	0.518 (.3)	**0.270** (.8)	**1.556** (.8)	**0.496** (.8)
PMTLM-DC (KL)	0.491 (.3)	1.865 (.3)	0.511 (.3)	0.406 (.3)	2.084 (.3)	**0.520** (.3)	0.260 (.8)	1.577 (.8)	0.492 (.8)

Table 4.2: The best normalized mutual information (NMI), variational of information (VI) and pairwise F-measure (PWF) achieved by each algorithm. Values marked by † are quoted from [56]; other values are based on our implementation. The best values are shown in bold; note that we seek to maximize NMI and PWF, and minimize VI. For PHITS-PLSA, PMTLM, and PMTLM-DC, the number in parentheses is the best value of the relative weight α of content vs. links. Refining the labeling returned by the EM algorithm with the Kernighan-Lin heuristic is indicated by (KL).

Results

The best NMI, VI, and PWF we observed for each algorithm are given in Table 4.2, where for LINK-LDA and C-PLDC we quote results from [56]. The metrics of NMI and PWF used in [56] are identical to ours. For algorithms with tunable parameters, including ours, PHITS-PLSA and RTM, we tuned them based on the entire data set in order to measure its best possible performance. Of course, in practice one would tune these parameters based on partial knowledge, such as the topics of a validation set of documents, and then use those parameter values to generalize to the test set.

We see that even without the additional step of local search, our algorithm does very well, outperforming all other methods we tried on Citeseer and PubMed and all but C-PLDC on Cora. (Note that we did not test LINK-LDA or C-PLDC on PubMed.) Degree correction (PMTLM-DC) improves accuracy significantly for PubMed.

Refining our labeling with the KL heuristic improved the performance of our algorithm significantly for Cora and Citeseer, giving us a higher accuracy than all the other methods we tested. For PubMed, local search did not increase accuracy in a statistically significant way. In fact, on some runs it decreased the accuracy slightly compared to the initial labeling $\hat{z}$ obtained from our EM algorithm; this is counterintuitive, but it shows that increasing the likelihood of a labeling in the model can decrease its accuracy.

In Figure 4.3, we show how the performance of PMTLM, PMTLM-DC, PHITS-PLSA varies as a function of α, the relative weight of content vs. links. Recall that at $\alpha = 0$ these algorithms label documents solely on the basis of their links, while at $\alpha = 1$ they only pay attention to the content. Each point consists of the top 20 runs with that value of α.

Figure 4.3 also shows that the optimal α and its sensitivity to performance differs between data sets. For Cora and Citeseer, there is an intermediate value of α at which

PMTLM and PMTLM-DC have the best accuracy. However, this peak is fairly broad, showing that we do not have to tune α very carefully. For PubMed, where we also normalized the content information by document length, PMTLM-DC performs best at a particular value of α.

We compare the running time of these algorithms, including PMTLM, PMTLM-DC with and without the KL heuristic, in Table 4.1. For algorithms with tunable parameters, we show the running time for a single value of that parameter. For our algorithms and PHITS-PLSA, we show the running time for $\alpha = 0.5$, giving the content and the links equal weight. We see that our EM algorithm is much faster than the variational EM algorithm for RTM, and is scalable in that it grows linearly with the size of the corpus.

4.3.4 Link prediction

Link prediction (e.g. [16, 36, 59]) is a natural generalization task in networks, and another way to measure the quality of our model and our EM algorithm. Based on a training set consisting of a subset of the links, our goal is to rank all pairs without an observed link according to the probability of a link between them. For our models, we rank pairs according to the expected number of links $A_{dd'}$ in the Poisson distribution, (4.2) and (4.3), which is monotonic in the probability that at least one link exists.

We can then predict links between those pairs where this probability exceeds some threshold. Since we are agnostic about this threshold and about the cost of Type I vs. Type II errors, we follow other work in this area by defining the accuracy of our model as the AUC, i.e. the probability that a random true positive link is ranked above a random true non-link. Equivalently, this is the area under the *receiver operating characteristic* curve (ROC). Our goal is to do better than the baseline AUC

of 1/2, corresponding to a random ranking of the pairs.

We carried out 10-fold cross-validation, in which the links in the original graph are partitioned into 10 subsets with equal size. For each fold, we use one subset as the test links, and train the model using the links in the other 9 folds. We evaluated the AUC on the held-out links and the non-links. For Cora and Citeseer, all the non-links are used. For PubMed, we randomly chose 10% of the non-links for comparison. We trained the models with the same settings as those for document classification in Section 4.3.3; we executed 100 independent runs for each test. Note that unlike the document classification task, here we used the full topic mixtures to predict links, not just the discrete labels consisting of the most-likely topic for each document.

Note that PMTLM-DC assigns S_d to be zero if the degree of d is zero. This makes it impossible for d to have any test link with others if its observed degree is zero in the training data. One way to solve this is to assign a small positive value to S_d even if d's degree is zero. Our approach assigns S_d to be the smallest value among those $S_{d'}$ that are non-zero.

Figure 4.4(a) gives the AUC values for PMTLM and PMTLM-DC as a function of the relative weight α of content vs. links. The green horizontal line in each of those subplots represent the highest AUC value achieved by the RTM model for each data set, using the best value of ρ among those specified in Section 4.3.3. Note that the optimal value of the tunable parameters is task-dependent: the optimal value ρ in RTM, or α in our algorithms and PHITS-PLSA, is not necessarily the same for link prediction as it is for document classification. Interestingly, for Cora and Citeseer the optimal value of α is smaller than in Figure 4.3, showing that content is less important for link prediction than for document classification. Thus, according to our experiments on both document classification and link prediction, the best choice of α depends not only on the data set, but also on the task.

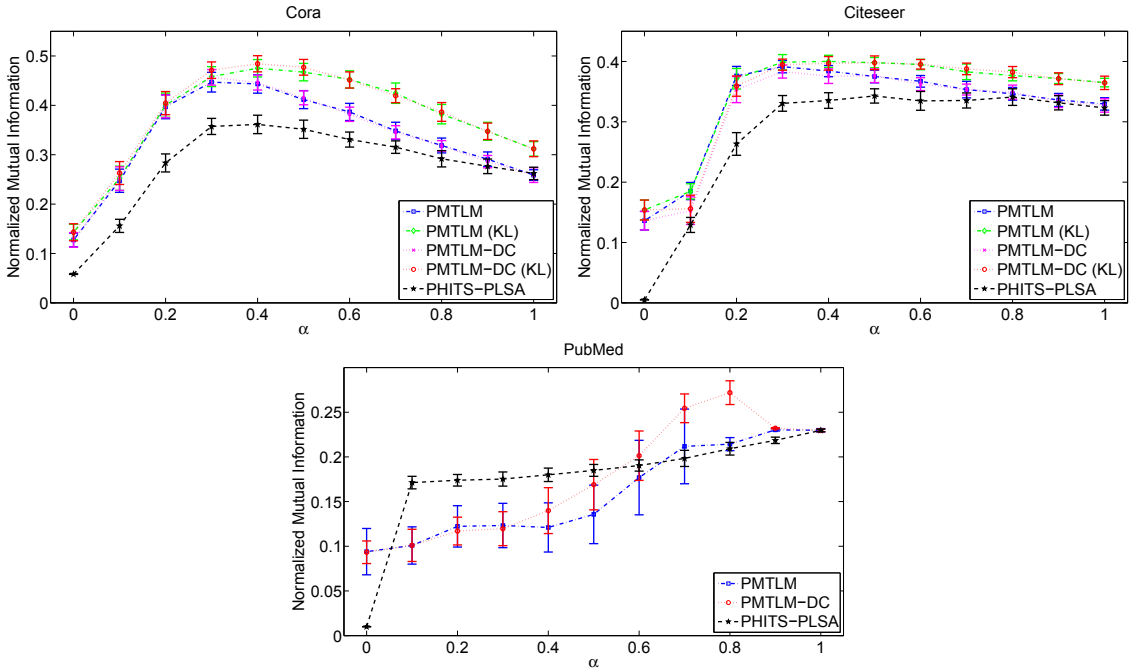
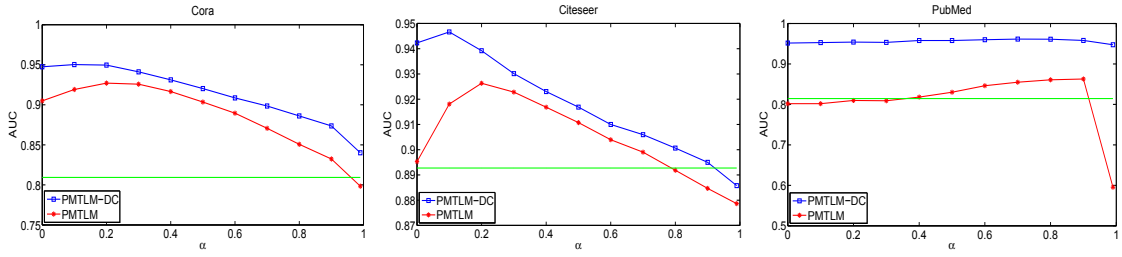
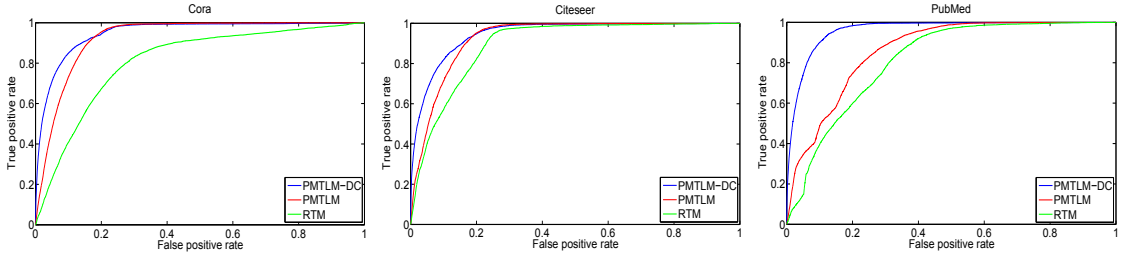


Figure 4.3: The accuracy of PMTLM, PMTLM-DC, and PHITS-PLSA on the document classification task, measured by the NMI, as a function of the relative weight α of the content vs. the links. At $\alpha = 0$ these algorithms label documents solely on the basis of their links, while at $\alpha = 1$ they pay attention only to the content. For Cora and Citeseer, there is a broad range of α that maximizes the accuracy. For PubMed, the degree-corrected model PMTLM-DC performs best at a particular value of α.

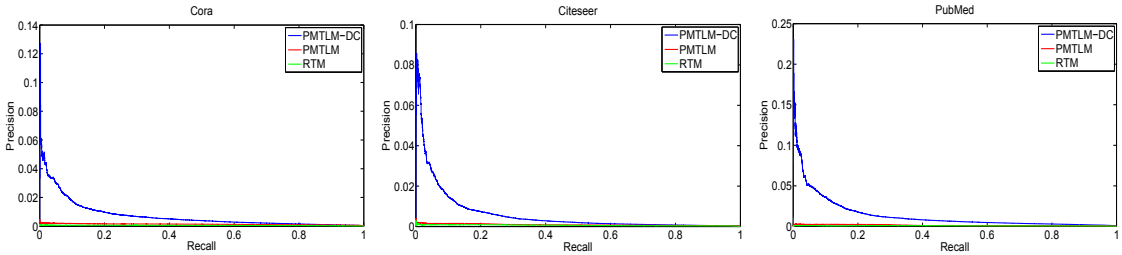
We also plot the *receiver operating characteristic* (ROC) curves and precision-recall curves that achieve the highest AUC values in Figure 4.4(b) and Figure 4.4(c) respectively. We see that, for all three data sets, our models outperform RTM, and that the degree-corrected model PMTLM-DC is significantly more accurate than the uncorrected one.



(a) AUC values for different α.



(b) ROC curves achieving the highest AUC values.



(c) Precision-recall curves achieving the highest AUC values.

Figure 4.4: Performance on the link prediction task. For all three data sets and all the α values, the PMTLM-DC model achieves higher accuracy than the PMTLM model. In contrast to Figure 4.3, for this task the optimal value of α is relatively small, showing that the content is less important, and the topology is more important, for link prediction than for document classification. The green line in Figure 4.4(a) indicates the highest AUC achieved by the RTM model, maximized over the tunable parameter ρ. Our models outperform RTM on all three data sets. In addition, the degree-corrected model (PMTLM-DC) does significantly better than the uncorrected version (PMTLM).

4.3.5 Keywords of the inferred topics

We also tested our algorithm on a patent citation network, which consists of patents related to "microprocessor" technologies. The network contains 1002 patents and

1640 links. By extracting the words of the titles and abstracts in the patent documents, stemming the words, filtering the stop words and the words that appeared in only one document, we get a vocabulary with 1692 words and a corpus of size 36814. We used the Porter stemming algorithm to derive the stems of the words. For example, the algorithm will reduce words "tested", "testing" and "tests" to the root word "test". The stop words include those common English words such as "a", "be" and "the". We also filtered out those corpus stop words which are common words in the whole corpus. These words include "microprocessor", "data", "system", "method" and so on. These corpus stop words have low tf-idf (term frequency–inverse document frequency) values. A word with a tf-idf value below some threshold will be filtered out.

We set the number of topics to 5. In Figure 4.5, we show how the document classification performance of PMTLM and PMTLM-DC varies as a function of the relative weight α. Both PMTLM and PMTLM-DC achieve the highest performance at $\alpha = 0.2$. The NMI values are computed based on the official patent classification, which is represented by a hierarchy of classes with two levels. We focus on the top level classes. For example, the class "710/5" and "710/113" belong to the same superclass of "710". A simple heuristic algorithm is applied to filter out those keywords that are associated with multiple topics. Those keywords which are mostly associated with only one topic given by the PMTLM model at $\alpha = 0.2$ are listed in Table 4.3. From these keywords, we can recognize that topic 1 is about data access and operations, topic 2 is about testing and debugging, topic 3 is about the power supply, topic 4 is about arithmetic and topic 5 is about control flow.

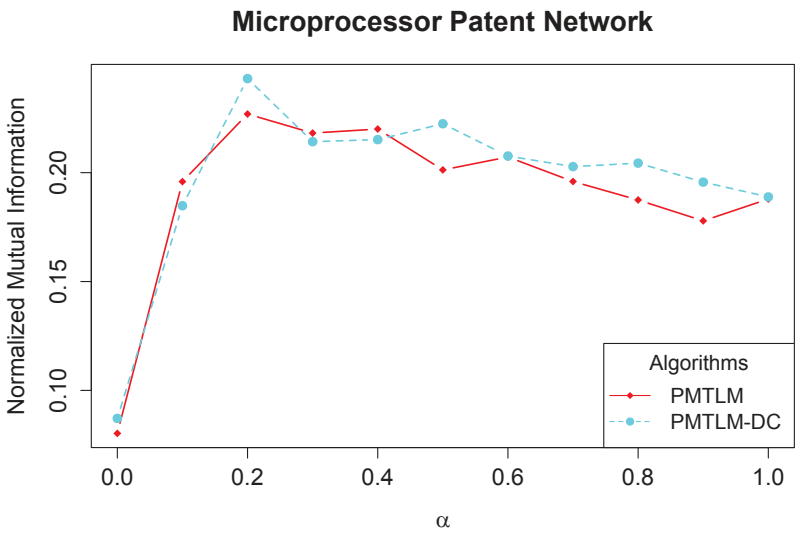


Figure 4.5: The accuracy of PMTLM and PMTLM-DC on document classification for the microprocessor patent network, measured by the NMIwith the official classification, as a function of the relative weight α of the content vs. the links.

Table 4.3: Top 10 words for 5 topics in the microprocessor patent network.

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
burst	test	power	arithmetic	instruct
coprocessor	debug	reset	multiplex	pipeline
intelligent	emulate	pulse	ALU	branch
asynchronous	develop	frequencies	microinstruction	superscalar
write-back	embedded	voltage	bidirectional	operand
attached	trace	consumption	single-chip	predict
transaction	secure	sense	microprogram	tag
DMA	in-circuit	drive	busses	subsequent
secondary	encrypted	adjust	microcontrol	dispatch
prioritization	model	interval	simplified	concurrence

4.4 Summary

Data that contains both links and content is very common in the real world, such as the document citation networks and online web pages, and a couple of models have

been developed in literatures to use this kind of data for document classification or link prediction. However these models have not achieved both high accuracy and high scalability. For example, the PHITS-PLSA model is scalable, but prone to overfitting, and the RTM model, which is the state-of-the-art model, is not scalable.

Our model combines PLSA and the BKN model. The simplicity of these models allows us to use both the content and links in a very efficient way; and the natural compatability of the PLSA and BKN model makes the combined model achieve high accuracy on both document classification and link prediciton. We developed a highly scalabe EM algorithm to infer the parameters, and the time complexity of the algorithm is linear in the size of the data set if we treat the number of topics as a constant.

We choose the uniform prior on both the topic mixtures and the word-topic distributions, i.e., those θ and β parameters. Although Dirichlet priors can be added and the MAP estimation doesn't increase the complexity of the algorithm, empirical results on real-world data sets show that the uniform prior gives a very impressive performance, out-performing other methods.

Balancing the content and links is a very important issue. In our models, we use content normalization and linear interpolation to achieve the balance. Content normalization makes documents of different sizes equally important; it also increases the likelihood for the content and this is especially helpful when the word counts is available, i.e., the C_{dw} is not limited to be binary. Linear interpolation lets us go from only caring about links to only caring about content. It's interesting that the optimal value of the relative weight α for text vs. links depends on both the data set and the task. Determining the optimal value of α automatically is tough but important. We leave this to future work.

Chapter 5

Conclusions

Community detection in complex networks is challenging as there are different kinds of community structure. Even in the same network, there may exist multiple community structures with quite different perperties. Stochastic block models are powerful tools for detecing communities based on the network topology. Based on the concept of statistical equivalence, stochastic block models are able to detect *functional communities* which may present assortative mixing, disassortative mixing or the mixture of both. Due to the natural complexity of the community structures in real world, such flexibility exhibited by stochastic block models is crucial. However, the overall simplicity of the original block models makes them hard to achieve satisfactory performance in many scenarios. In resent years, research focuses on developing extended models to incorporate more information so that the new models are able to work for those real world applications. My research on block models focuses on understanding the underlying assumptions made by the models, the strengths and weaknesses of the models, and developing new models to overcome those weaknesses and adapt them to new applications.

The lesson we have learned is that *a generative model can only learn from the data*

that it is required to generate. The degree-corrected block model provides a powerful tool for dealing with networks with inhomogeneous degree distributions. However, since degree-corrected models are given the vertex degrees as parameters and are under no obligation to explain them, they cannot use degrees to help them classify vertices. We have introduced the oriented degree-corrected (ODC) model and degree generation (DG) to address this problem. The ODC and DG models allow for broad or heavy-tailed degree distributions, while still being able to take vertex degrees into account when inferring communities. Another limitation of the stochastic block models is that these models classify nodes based on the network topology alone because only the links are generated. This makes the models unable to use the rich attributes of the nodes for community detection. We have developed the Poisson mixed-topic link model (PMTLM) which generates both links and text.

The ODC model is forced to generate edge orientations. Unlike the directed degree-corrected (DDC) block model, which takes both in- and out-degrees as parameters, ODC is able to capture certain correlations between the in- and out-degrees. Simply put, for ODC, two vertices are unlikely to be in the same community if one has high in-degree and low out-degree while another has high out-degree and low in-degree. If the network is highly directed or asymmetric, the edge orientations can help ODC find community structures that DDC fails to perceive.

When applying these models on directed networks, DC totally ignore the direction of the edges and generate the undirected version of the given network. DC prefers community structures where the node total degrees are highly skewed in each community. DDC generates the directed network completely, thus both the in- and out-degrees are corrected. In other words, DDC prefers community structures where in each community both the in- and out-degrees are highly skewed. ODC generates the undirected network first using the DC model and then generates the edge orientations. ODC prefers disassortative mixing and highly directed inter-community

connections.

Non-degree-corrected and degree-corrected block models represents two extremes, one prefers homogeneous degree distributions in each community and the other prefers inhomogeneous degree distributions. Degree generation (DG) fills the gap between them. We consider degree-generated block models. These models use degree-corrected block models as a subroutine, but they first generate the expected degree of each vertex from a prior distribution in each community. DG works jointly with degree-corrected block models, like the DC, DDC or ODC, so that the joint model is able to use the prior knowledge of the degree distributions in each community, which can be homogeneous, inhomogeneous or any parameterized distribution families. DG models can achieve high accuracy even when the density of connections within or between communities is close to uniform, this is illustrated in synthetic networks for small λ in Section 3.3.1. Augmenting block models, such as ODC, with degree generation also appears to speed up their convergence in some cases, helping simple algorithms like MCMC handle large networks without the benefit of expensive preprocessing steps like the KL heuristic. However, the effectiveness of DG depends heavily on knowing the correct form of the degree distribution in each community.

With all these variants of the block model, ranging from the "classic" version to degree-corrected and degree-generated variants, we now have a wide variety of tools for inferring structure in network data. Each model will perform better on some networks and worse on others. A better understanding of the strengths and weaknesses of each one—which kinds of structure they can see and which they are blind to—will help us select the right algorithm each time we meet a new network.

All the above models focus on community detection in networks by the network topology alone. Sometime topological information might be insufficient to the learning task. In that case the nodes' attributes, such as the nodes' location, demographic variables, or (in document networks) their content, will be extremely useful for some

applications. As illustrated in Section 4.3.3, when we use the link information solely, we failed to detect the document topics on those document-citation networks. This is a lesson we learned from these real-world networks which have very sparse pairwise connections and thus the network topology itself doesn't present detectable community structures.

We have introduced a new generative model, the Poisson mixed-topic link model (PMTLM), for topic detection or link prediction in document networks. In this model, both the pairwise links and the document content are observations. The model is requested to generate both links and words so that it can use both the link and text information. The new model is a marriage between the text-based Probabilistic Latent Semantic Analysis (PLSA) [30] model and the link-based Ball-Karrer-Newman (BKN) mixed membership block model [7]. Because of its mathematical simplicity (compared with the models like RTM), its parameters can be inferred with a particularly simple and efficient EM algorithm. Our experiments on document classification and link prediction show that it achieves high accuracy and scalability for a variety of data sets, outperforming other methods.

Chapter 6

Future Work

6.1 Model selection

In Chapter 3, we introduced a bunch of stochastic block models. In my work, the strengths and weaknesses of these models are carefully analyzed. Section 3.4 gives a summary. This provides us a guide to select a proper model based on the knowledge about the community structure we are looking for. However in many scenarios, especially at the exploration stage, we just have no idea about the community structure at all. All we have is the network data itself. In that case, we want to choose the model based on the data alone. This is a model selection problem, which is a very important topic in statistics and machine learning research.

In Chapter 4, the Poisson mixed-topic link model also has two versions, and for each version, we have a tunable parameter α to be determined. Choosing the right version of the model and the best value of α is also a model selection problem. We are assuming that the number of communities are known, but this is not true in many real-world applications. Choosing the best value of community number is again a model selection problem. We leave all these challenging problems to our future

work and we believe different methods should be explored for different problems. Some possible solutions include the information criterion method [3, 49], minimum description length, hypothese testing, Bayesian model averaging, holdout and cross-validation and so on.

6.2 Active learning

Our previous work in [42] describes an active learning method for community detection in networks. The underlying model used is a variant of the non-degree-corrected stochastic block model. It will be interesting to try degree-corrected block models, including DC, DDC and ODC, for applications in which the degree-correction will benefit us. The previous method relies on a MCMC Gibbs sampler, which makes it hard to adapt this active learning method to the Poisson mixed-topic link models where an efficient EM algorithm is used. That method is also not very scalable since it needs many independent samples to estimate the mutual information. Exploring new active learning methods that can work with the EM algorithm or designing a MCMC algorithm for the mixed-topic link models will be two possible directions. The first direction will be more appealing due to its scalability.

Appendices

Appendix A

Maximum likelihood estimates of the DDC model

We maximize the log-likelihood function (3.4),

$$\begin{aligned}\log P(G \,|\, S, \omega, g) = &\sum_u (d_u^{\text{out}} \log S_u^{\text{out}} + d_u^{\text{in}} \log S_u^{\text{in}}) \\ &+ \sum_{rs} (m_{rs} \log \omega_{rs} - \kappa_r^{\text{out}} \kappa_s^{\text{in}} \omega_{rs}) \,, \end{aligned} \tag{A.1}$$

where we have imposed the constraints on the S parameters

$$\sum_{u:g_u=r} S_u^{\text{out}} = \kappa_r^{\text{out}} \quad \text{and} \quad \sum_{u:g_u=r} S_u^{\text{in}} = \kappa_r^{\text{in}} \,. \tag{A.2}$$

For each block r, we associate Lagrange multipliers $\lambda_r^{\text{out}}, \lambda_r^{\text{in}}$ with these constraints. For each vertex u, taking the partial derivative of the log-likelihood with respect to S_u^{out} and S_u^{in} gives

$$\frac{d_u^{\text{out}}}{S_u^{\text{out}}} = \lambda_{g_u}^{\text{out}} \quad \text{and} \quad \frac{d_u^{\text{in}}}{S_u^{\text{in}}} = \lambda_{g_u}^{\text{in}} \,. \tag{A.3}$$

To satisfy the constraints (A.2), we take $\lambda_r^{\text{out}} = \lambda_r^{\text{in}} = 1$ for all r, so that

$$\hat{S}_u^{\text{out}} = d_u^{\text{out}} \quad \text{and} \quad \hat{S}_u^{\text{in}} = d_u^{\text{in}} \,. \tag{A.4}$$

Setting the partial derivative of the log-likelihood function with respect to ω_{rs} to zero then gives

$$\hat{\omega}_{rs} = \frac{m_{rs}}{\kappa_r^{\text{out}} \kappa_s^{\text{in}}} \,. \tag{A.5}$$

Appendix B

Another view of the ODC model

Here we show that the oriented degree-corrected (ODC) model is a special case of the directed degree-corrected (DDC) model. Recall that the ODC model first generates an undirected graph according to the DC model with parameters S_u and ω_{rs}, and then orients each edge (u, v) from u to v with probability ρ_{g_u,g_v}. The number of directed edges from u to v is then Poisson-distributed as

$$A_{uv} \sim \mathrm{Poi}(S_u S_v \omega_{g_u,g_v} \rho_{g_u,g_v}) \,. \tag{B.1}$$

But if we write

$$\omega'_{rs} = \omega_{rs}\rho_{rs} \,, \tag{B.2}$$

then

$$A_{uv} \sim \mathrm{Poi}(S_u S_v \omega'_{g_u,g_v}) \,. \tag{B.3}$$

Thus ODC is the special case of DDC where $S_u^{\mathrm{in}} = S_u^{\mathrm{out}} = S_u$ for all vertices u.

For completeness, we check that the two models correspond when we set these parameters equal to their MLEs. We impose the constraint $\sum_{u:g_u=r} S_u = \kappa_r =$

$\kappa_r^{\text{out}} + \kappa_r^{\text{in}}$ for all blocks r. Ignoring constants, the log-likelihood is then

$$\log P(G \,|\, S, \omega', g) = \sum_u d_u \log S_u + \sum_{rs} (m_{rs} \log \omega'_{rs} - \kappa_r \kappa_s \omega'_{rs}) \,, \tag{B.4}$$

where $d_u = d_u^{\text{out}} + d_u^{\text{in}}$. The MLEs for S_u and ω'_{rs} are then

$$\hat{S}_u = d_u \,, \quad \hat{\omega}'_{rs} = \frac{m_{rs}}{\kappa_r \kappa_s} \,. \tag{B.5}$$

Thus $\hat{\omega}'_{rs} = \hat{\omega}_{rs} \hat{\rho}_{rs}$ where

$$\hat{\omega}_{rs} = \frac{\bar{m}_{rs}}{\kappa_r \kappa_s} \quad \text{and} \quad \hat{\rho}_{rs} = \frac{m_{rs}}{\bar{m}_{rs}} \,, \tag{B.6}$$

recovering (3.11).

Appendix C

Bayesian estimation for DG models

Bayesian inference focuses on posterior distributions of parameters rather than on point estimates. In hierarchical models like DG-DDC, the full Bayesian posterior of the S parameters (omitting the other parameters g and ω) is

$$P(S \mid G) = \int P(S \mid G, \psi)\, P(\psi \mid G)\, \mathrm{d}\psi\,. \tag{C.1}$$

Here we employ the Empirical Bayesian method, and use point estimates for the hyperparameters ψ, namely their MLEs $\hat{\psi}$,

$$\begin{aligned} \hat{\psi} &= \operatorname*{argmax}_{\psi} P(G \mid \psi) \\ &= \operatorname*{argmax}_{\psi} \int P(G \mid S, \psi)\, P(S \mid \psi)\, \mathrm{d}S\,. \end{aligned} \tag{C.2}$$

With this approximation we have

$$\begin{aligned} P(S \,|\, G) &\approx P(S \,|\, G, \hat{\psi}) \\ &= \frac{P(G \,|\, S)\, P(S \,|\, \hat{\psi})}{P(G \,|\, \hat{\psi})} \\ &= \frac{P(G \,|\, S)\, P(S \,|\, \hat{\psi})}{\int P(G \,|\, S, \hat{\psi})\, P(S \,|\, \hat{\psi})\, \mathrm{d}S}\,, \end{aligned} \tag{C.3}$$

where we used Bayes' rule in the second line.

Computing the posterior $P(S \,|\, G)$ is usually difficult, as the integral in the denominator of (C.3) is often intractable. However, with a clever choice of the prior distribution $P(S \,|\, \psi)$, we can work out an analytic solution. It is called the *conjugate prior* of the likelihood term. We focus here on DG-DDC; the calculations for other degree-generated models are similar.

Say that a random variable X is Gamma-distributed with parameters α, β, and write $X \sim \Gamma(\alpha, \beta)$, if its probability distribution is

$$f(x; \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}\, x^{\alpha-1}\, \mathrm{e}^{-\beta x}\,. \tag{C.4}$$

In DG-DDC, the likelihood (3.3) can be written (where we have plugged in the MLEs for ω, and substituted $\kappa_r^{\text{out}} = \sum_{u:g_u=r} d_u^{\text{out}}$)

$$P(G \,|\, S^{\text{out}}) = \frac{\prod_u (S^{\text{in}})^{d_u^{\text{in}}} \prod_{rs} \omega_{rs}^{m_{rs}}}{\prod_{uv} A_{uv}!} \prod_u (S_u^{\text{out}})^{d_u^{\text{out}}} \exp\bigl(-S_u^{\text{out}}\bigr)\,. \tag{C.5}$$

If we assume that the S^{in} and S^{out} for each u are independent, this is proportional to a product of Gamma distributions with parameters $\alpha = d_u^{\text{out}} + 1$ and $\beta = 1$ for each S_u^{out}.

A natural conjugate prior for Gamma distributions is the Gamma distribution itself. Let the hyperparameters ψ_r^{out} for each block r consist of a pair $(\alpha_r^{\text{out}}, \beta_r^{\text{out}})$, and consider the prior

$$S_u^{\text{out}} \sim \Gamma(\alpha_{g_u}^{\text{out}}, \beta_{g_u}^{\text{out}})\,. \tag{C.6}$$

That is,

$$P(S_u^{\text{out}} \,|\, \psi_{g_u}^{\text{out}}) = \frac{(\beta_{g_u}^{\text{out}})^{\alpha_{g_u}^{\text{out}}}}{\Gamma(\alpha_{g_u}^{\text{out}})} (S_u^{\text{out}})^{\alpha_{g_u}^{\text{out}}-1} \exp(-\beta_{g_u}^{\text{out}} S_u^{\text{out}}) \,, \tag{C.7}$$

Multiplying this prior by the likelihood (C.5) stays within the family of Gamma distributions, and simply updates the parameters:

$$\begin{aligned} P(S_u^{\text{out}} \,|\, G) &\propto P(S_u^{\text{out}} \,|\, \psi_{g_u}^{\text{out}})\, P(G \,|\, S^{\text{out}}) \\ &\propto (S_u^{\text{out}})^{\alpha_{g_u}^{\text{out}} + d_u^{\text{out}} - 1} \exp\big(-S_u^{\text{out}} \left(\beta_{g_u}^{\text{out}} + 1\right)\big) \,. \end{aligned} \tag{C.8}$$

Thus the posterior distribution is

$$S_u^{\text{out}} \sim \Gamma\big(\alpha_{g_u}^{\text{out}} + d_u^{\text{out}}, \beta_{g_u}^{\text{out}} + 1\big) \,. \tag{C.9}$$

Note that if we use a uninformative prior, i.e., in the limit $\alpha_{g_u}^{\text{out}} = 1$ and $\beta_{g_u}^{\text{out}} = 0$, the Gamma prior reduces to a uniform prior. The maximum a posteriori (MAP) estimate of S_u^{out} is

$$\hat{S}_u^{\text{out}} = d_u^{\text{out}} \,, \tag{C.10}$$

and similarly for S_u^{in}, just as we obtained for the MLEs in (3.5).

However, our goal is to integrate over S, not focus on its MAP estimate. So let us continue the Bayesian analysis. Assuming the S parameters are independent, then their joint posterior is simply a product of their individual posteriors

$$\begin{aligned} P(S|G) &= \prod_u P(S_u^{\text{out}}|G) P(S_u^{\text{in}}|G) \\ &= \prod_u f\big(S_u^{\text{out}}; \alpha_{g_u}^{\text{out}} + d_u^{\text{out}}, \beta_{g_u}^{\text{out}} + 1\big)\, f\big(S_u^{\text{in}}; \alpha_{g_u}^{\text{in}} + d_u^{\text{in}}, \beta_{g_u}^{\text{in}} + 1\big) \,. \end{aligned} \tag{C.11}$$

Then we can calculate the integral in (C.2) and (C.3) by the simple algebra:

$$\begin{aligned}
&\int P(G \mid S, \psi) P(S \mid \psi) \, \mathrm{d}S = \frac{P(G \mid S) P(S \mid \psi)}{P(S \mid G)} \\
=&\frac{\prod_u f(S_u^{\text{out}}; d_u^{\text{out}} + 1, 1) \, f(S_u^{\text{in}}; d_u^{\text{in}} + 1, 1) \, f(S_u^{\text{out}}; \alpha_{g_u}^{\text{out}}, \beta_{g_u}^{\text{out}}) \, f(S_u^{\text{in}}; \alpha_{g_u}^{\text{in}}, \beta_{g_u}^{\text{in}})}{\prod_u f\left(S_u^{\text{out}}; \alpha_{g_u}^{\text{out}} + d_u^{\text{out}}, \beta_{g_u}^{\text{out}} + 1\right) f\left(S_u^{\text{in}}; \alpha_{g_u}^{\text{in}} + d_u^{\text{in}}, \beta_{g_u}^{\text{in}} + 1\right)} \\
=&\frac{\prod_u {\beta_{g_u}^{\text{out}}}^{\alpha_{g_u}^{\text{out}}} {\beta_{g_u}^{\text{in}}}^{\alpha_{g_u}^{\text{in}}} \Gamma\left(\alpha_{g_u}^{\text{out}} + d_u^{\text{out}}\right) \Gamma\left(\alpha_{g_u}^{\text{in}} + d_u^{\text{in}}\right)}{\prod_u \left(\beta_{g_u}^{\text{out}} + 1\right)^{\alpha_{g_u}^{\text{out}} + d_u^{\text{out}}} \left(\beta_{g_u}^{\text{in}} + 1\right)^{\alpha_{g_u}^{\text{in}} + d_u^{\text{in}}} \Gamma(d_u^{\text{out}} + 1) \, \Gamma(d_u^{\text{in}} + 1) \, \Gamma\left(\alpha_{g_u}^{\text{out}}\right) \Gamma\left(\alpha_{g_u}^{\text{in}}\right)} .
\end{aligned} \tag{C.12}$$

Now that the dependence of the numerator and denominator on S has cancelled out, the integral is a function only of the hyperparameters ψ, making it possible to do the point estimate of ψ in (C.2). In our case, optimizing for $\hat{\psi}$ requires some numeric techniques, but it is nonetheless doable.

Empirical Bayesian solution not only gives better approximation to the original problem, it also make it possible to integrate prior knowledge if available. On top of that, because the posterior is now a direct function of the hyperparameters ψ, we no longer have to worry about the Poisson noise when estimating ψ indirectly from degrees.

On a final note, the above result only holds for Gamma priors. With any other prior, the integral may not be this simple.

Appendix D

Power-law distribution with upper bound

In this section, we show that imposing an upper bound on our power-law distributions in order to ensure a certain average degree does not appreciably change the procedure of [15] for estimating the exponent. Suppose x is distributed as a power law lower bound $x_{\min}$, upper bound $x_{\max}$, and exponent $\alpha > 0$. Then

$$p(x) = \frac{\alpha - 1}{x_{\min}^{1-\alpha} - x_{\max}^{1-\alpha}} \, x^{-\alpha}, \quad x_{\min} \leq x \leq x_{\max} \,. \tag{D.1}$$

Given a random sample $\mathbf{x} = \{x_1, \ldots, x_n\}$ drawn from this distribution independently, the likelihood function is

$$p(\mathbf{x}) = \prod_{i=1}^{n} \frac{\alpha - 1}{x_{\min}^{1-\alpha} - x_{\max}^{1-\alpha}} \, x_i^{-\alpha} = \left(\frac{\alpha - 1}{x_{\min}^{1-\alpha} - x_{\max}^{1-\alpha}} \right)^n \prod_{i=1}^{n} x_i^{-\alpha} \,. \tag{D.2}$$

Thus, the log-likelihood is

$$\log p(\mathbf{x}) = n \left(\log(\alpha - 1) - \log \left(x_{\min}^{1-\alpha} - x_{\max}^{1-\alpha} \right) \right) - \alpha \sum_{i=1}^{n} \log x_i \,. \tag{D.3}$$

Taking the derivative with respect to α gives

$$\frac{\partial \log p(\mathbf{x})}{\partial \alpha} = n \left(\frac{1}{\alpha - 1} + \frac{x_{\min}^{1-\alpha} \log x_{\min} - x_{\max}^{1-\alpha} \log x_{\max}}{x_{\min}^{1-\alpha} - x_{\max}^{1-\alpha}} \right) - \sum_{i=1}^{n} \log x_i \,. \tag{D.4}$$

Setting (D.4) to zero, we get

$$\frac{1}{\alpha-1}+\frac{x_{\min}^{1-\alpha}\log x_{\min}-x_{\max}^{1-\alpha}\log x_{\max}}{x_{\min}^{1-\alpha}-x_{\max}^{1-\alpha}}=\frac{\sum_{i=1}^{n}\log x_i}{n}\,. \tag{D.5}$$

If $x_{\min}=1$ and $x_{\max}\to\infty$, then solving (D.5) gives the MLE for α just as in (3.17).

Appendix E

Update equations for PMTLM

In this appendix, we derive the update equations (4.10)–(4.12) for the parameters η, β, and θ, giving the M step of our algorithm.

Recall that the likelihood is given by (4.7) and (4.8). For identifiability, we impose the normalization constraints

$$\forall z : \sum_w \beta_{zw} = 1 \tag{E.1}$$

$$\forall d : \sum_z \theta_{dz} = 1 \tag{E.2}$$

For each topic z, taking the derivative of the likelihood with respect to η_z gives

$$0 = \frac{1}{1-\alpha}\frac{\partial \mathcal{L}}{\partial \eta_z} = \frac{1}{\eta_z}\sum_{dd'} A_{dd'} q_{dd'}(z) - \sum_{dd'} \theta_{dz}\theta_{d'z}\,. \tag{E.3}$$

Thus

$$\eta_z = \frac{\sum_{dd'} A_{dd'} q_{dd'}(z)}{\sum_{dd'} \theta_{dz}\theta_{d'z}} = \frac{\sum_{dd'} A_{dd'} q_{dd'}(z)}{\left(\sum_d \theta_{dz}\right)^2}\,. \tag{E.4}$$

Plugging this in to (4.8) makes the last term a constant, $-1/2\sum_{dd'} A_{dd'} = -M$. Thus we can ignore this term when estimating θ_{dz}.

Similarly, for each topic z and each word w, taking the derivative with respect to β_{zw} gives

$$\nu_z = \frac{1}{\alpha}\frac{\partial \mathcal{L}}{\partial \beta_{zw}} = \frac{1}{\beta_{zw}} \sum_d \frac{1}{L_d} C_{dw} h_{dw}(z)\,, \tag{E.5}$$

where ν_z is the Lagrange multiplier for (E.1). Normalizing β_z determines ν_z, and gives

$$\beta_{zw} = \frac{\sum_d (1/L_d) C_{dw} h_{dw}(z)}{\sum_d (1/L_d) \sum_{w'} C_{dw'} h_{dw'}(z)}\,. \tag{E.6}$$

Finally, for each document d and each topic z, taking the derivative with respect to θ_{dz} gives

$$\lambda_d = \frac{\partial \mathcal{L}}{\partial \theta_{dz}} = \frac{\alpha}{L_d \theta_{dz}} \sum_w C_{dw} h_{dw}(z) + \frac{1-\alpha}{\theta_{dz}} \sum_{d'} A_{dd'} q_{dd'}(z)\,, \tag{E.7}$$

where λ_d is the Lagrange multiplier for (E.2). Normalizing θ_d determines λ_d and gives

$$\theta_{dz} = \frac{(\alpha/L_d) \sum_w C_{dw} h_{dw}(z) + (1-\alpha) \sum_{d'} A_{dd'} q_{dd'}(z)}{\alpha + (1-\alpha)\kappa_d}\,. \tag{E.8}$$

Appendix F

Update equations for PMTLM-DC

Recall that in the degree-corrected model PMTLM-DC, the number of links between each pair of documents d, d' is Poisson-distributed with mean

$$S_d S_{d'} \sum_z \eta_z \theta_{dz} \theta_{d'z} \,. \tag{F.1}$$

To make the model identifiable, in addition to (E.1) and (E.2), we impose the following constraint on the degree-correction parameters,

$$\forall z : \sum_d S_d \theta_{dz} = 1 \,. \tag{F.2}$$

With this constraint, we have

$$\begin{aligned}
\mathcal{L} = {} & \alpha \sum_d \frac{1}{L_d} \sum_{wz} C_{dw} h_{dw}(z) \log \frac{\theta_{dz} \beta_{zw}}{h_{dw}(z)} \\
& + (1-\alpha) \sum_d \kappa_d \log S_d \\
& + \frac{1-\alpha}{2} \sum_{dd'z} \left(A_{dd'} q_{dd'}(z) \log \frac{\eta_z \theta_{dz} \theta_{d'z}}{q_{dd'}(z)} - S_d S_{d'} \eta_z \theta_{dz} \theta_{d'z} \right) .
\end{aligned} \tag{F.3}$$

The update equation (E.6) for β remains the same, since the degree-correction only affects the part of the model that generates the links, not the words. We now derive the update equations for η, S, and θ.

For each topic z, taking the derivative of the likelihood with respect to η_z gives

$$\begin{aligned} 0 = \frac{2}{1-\alpha}\frac{\partial L}{\partial \eta_z} &= \frac{1}{\eta_z}\sum_{dd'} A_{dd'} q_{dd'}(z) - \sum_{dd'} S_d S_{d'} \theta_{dz} \theta_{d'z} \\ &= \frac{1}{\eta_z}\sum_{dd'} A_{dd'} q_{dd'}(z) - 1\,, \end{aligned} \tag{F.4}$$

where we used (F.2). Thus

$$\eta_z = \sum_{dd'} A_{dd'} q_{dd'}(z)\,, \tag{F.5}$$

so η_z is simply the expected number of links caused by topic z. In particular,

$$\sum_z \eta_z = \sum_{dd'} A_{dd'} = \sum_d \kappa_d = 2M\,. \tag{F.6}$$

For S_d, we have

$$\begin{aligned} \frac{1}{1-\alpha}\frac{\partial L}{\partial S_d} &= \frac{\kappa_d}{S_d} - \sum_{d'z} S_{d'} \eta_z \theta_{dz} \theta_{d'z} \\ &= \frac{\kappa_d}{S_d} - \sum_z \eta_z \theta_{dz} = \sum_z \xi_z \theta_{dz}\,, \end{aligned} \tag{F.7}$$

where ξ_z is the Lagrange multiplier for (F.2). Thus

$$S_d = \frac{\kappa_d}{\sum_z (\eta_z + \xi_z)\theta_{dz}}\,. \tag{F.8}$$

We will determine ξ_z below. However, note that multiplying both sides of (F.7) by S_d, summing over d, and applying (F.2) and (F.6) gives

$$\sum_z \xi_z = 0\,. \tag{F.9}$$

Most importantly, for θ we have

$$\begin{aligned}\frac{\partial L}{\partial \theta_{dz}} &= \frac{1}{\theta_{dz}} \left(\frac{\alpha}{L_d} \sum_w C_{dw} h_{dw}(z) + (1-\alpha) \sum_{d'} A_{dd'} q_{dd'}(z) \right) \\ &\quad - (1-\alpha) \sum_{d'} S_d S_{d'} \eta_z \theta_{d'z} \\ &= \frac{1}{\theta_{dz}} \left(\frac{\alpha}{L_d} \sum_w C_{dw} h_{dw}(z) + (1-\alpha) \sum_{d'} A_{dd'} q_{dd'}(z) \right) \\ &\quad - (1-\alpha) S_d \eta_z \\ &= \lambda_d + (1-\alpha) S_d \xi_z \,, \end{aligned} \tag{F.10}$$

where λ_d is the Lagrange multiplier for (E.2), and where we applied (F.2) in the second equality. Multiplying both sides of (F.10) by θ_{dz}, summing over z, and applying (F.8) gives

$$\lambda_d = \alpha \,. \tag{F.11}$$

Summing over d and applying (F.2), (F.5), and (F.11) gives

$$\begin{aligned}\frac{1-\alpha}{\alpha} \xi_z &= \sum_d \frac{1}{L_d} \sum_w C_{dw} h_{dw}(z) - \sum_d \theta_{dz} \\ &= \sum_d \frac{1}{L_d} \sum_w C_{dw} \left(h_{dw}(z) - \theta_{dz} \right) \,. \end{aligned} \tag{F.12}$$

Thus ξ_z measures how the inferred topic distributions of the words $h_{dw}(z)$ differ from the topic mixtures θ_{dz}.

Finally, (F.10) and (F.11) give

$$\theta_{dz} = \frac{(\alpha/L_d) \sum_w C_{dw} h_{dw}(z) + (1-\alpha) \sum_{d'} A_{dd'} q_{dd'}(z)}{\alpha + (1-\alpha)(\eta_z + \xi_z) S_d} \,, \tag{F.13}$$

where η_z and ξ_z are given by (F.5) and (F.12).

Appendix G

Update Equations with Dirichlet Prior

If we impose a Dirichlet prior on θ, with parameters $\{\gamma_z\}$ for each topic z, this gives an additional term $\sum_{dz}(\gamma_z - 1)\log\theta_{dz}$ in the log-likelihood of both the PMTLM and PMTLM-DC models. This is equivalent to introducing pseudocounts $t_z = \gamma_z - 1$ for each z, which we can think of as additional words or links that we know are due to topic z. Our original models, without this term, correspond to the uniform prior with $\gamma_z = 1$ and $t_z = 0$. However, as long as $\gamma_z \geq 1$ so that the pseudocounts are nonnegative, we can infer the parameters of our model in the same way with no loss of efficiency.

In the PMTLM model, (E.8) becomes

$$\theta_{dz} = \frac{t_z + (\alpha/L_d)\sum_w C_{dw}h_{dw}(z) + (1-\alpha)\sum_{d'} A_{dd'}q_{dd'}(z)}{\sum_z t_z + \alpha + (1-\alpha)\kappa_d}\,. \tag{G.1}$$

In the degree-corrected model PMTLM-DC, (F.11) and (F.12) become

$$\lambda_d = \alpha + \sum_z t_z \tag{G.2}$$

and

$$\frac{1-\alpha}{\alpha}\,\xi_z = \sum_d \frac{1}{L_d} \sum_w C_{dw}\,(h_{dw}(z) - \theta_{dz}) + \frac{1}{\alpha}\sum_d \left(t_z - \theta_{dz} \sum_{z'} t_{z'} \right) . \tag{G.3}$$

Note that ξ_z has two contributions. One measures, as before, how the inferred topic distributions of the words $h_{dw}(z)$ differ from the topic mixtures θ_{dz}, and the other measures how the fraction $t_z / \sum_{z'} t_{z'}$ of pseudocounts for topic z differs from θ_{dz}.

Finally, (F.13) becomes

$$\theta_{dz} = \frac{t_z + (\alpha/L_d) \sum_w C_{dw} h_{dw}(z) + (1-\alpha) \sum_{d'} A_{dd'} q_{dd'}(z)}{\alpha + (1-\alpha)(\eta_z + \xi_z) S_d + \sum_{z'} t_{z'}} , \tag{G.4}$$

where η_z and ξ_z are given by (F.5) and (G.3).

References

[1] L. A. Adamic and N. Glance. The political blogosphere and the 2004 US election: divided they blog. *Proceedings of the 3rd international workshop on Link discovery*, pages 36–43, 2005.

[2] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2008.

[3] Hirotugu Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716–723, 1974.

[4] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74:47–97, 2002.

[5] Stefano Allesina and Mercedes Pascual. Food web models: a plea for groups. *Ecology letters*, 12(7):652–662, 2009.

[6] Carolyn J Anderson, Stanley Wasserman, and Katherine Faust. Building stochastic blockmodels. *Social Networks*, 14(1):137–161, 1992.

[7] B. Ball, B. Karrer, and M. E. J. Newman. Efficient and principled method for detecting communities in networks. *Phys. Rev. E*, 84(3):036103, 2011.

[8] S. Basu. *Semi-supervised Clustering: Probabilistic Models, Algorithms and Experiments*. PhD thesis, Department of Computer Sciences, University of Texas at Austin, 2005.

[9] Heiko Bauke, Cristopher Moore, Jean-Baptiste Rouquier, and David Sherrington. Topological phase transition in a network model with preferential attachment and node removal. *The European Physical Journal B*, 83(4):519–524, 2011.

[10] Peter J Bickel and Aiyou Chen. A nonparametric view of network models and Newman–Girvan and other modularities. *Proceedings of the National Academy of Sciences*, 106(50):21068–21073, 2009.

[11] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[12] Jonathan Chang and David M. Blei. Relational topic models for document networks. In *Proc. of Conf. on AI and Statistics (AISTATS)*, 2009.

[13] Jonathan Chang and David M. Blei. Hierarchical relational models for document networks. *Annals of Applied Statistics*, 4(1):124–150, 2010.

[14] Aiyou Chen, Arash A. Amini, Peter J. Bickel, and Elizaveta Levina. Fitting community models to large sparse networks. *CoRR*, abs/1207.2340, 2012.

[15] A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Review*, 51:661–703, 2009.

[16] Aaron Clauset, Cristopher Moore, and M. E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98–101, 2008.

[17] David Cohn and Huan Chang. Learning to Probabilistically Identify Authoritative Documents. In *International Conference on Machine Learning (ICML)*, 2000.

[18] David Cohn and Thomas Hofmann. The missing link-a probabilistic model of document content and hypertext connectivity. In *Neural Information Processing Systems (NIPS)*, 2001.

[19] Leon Danon, Albert Díaz-Guilera, Jordi Duch, and Alex Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(09):P09008, 2005.

[20] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Phys. Rev. E*, 84(6), 2011.

[21] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová. Inference and Phase Transitions in the Detection of Modules in Sparse Networks. *Phys. Rev. Lett.*, 107(6):065701, 2011.

[22] Elena Erosheva, Stephen Fienberg, and John Lafferty. Mixed-membership models of scientific publications. *Proc. National Academy of Sciences*, 101 Suppl:5220–7, 2004.

[23] S. E. Fienberg and S. S. Wasserman. Categorical data analysis of single sociometric relations. *Sociological methodology*, 12:156–192, 1981.

[24] Andrew T Fiore and Judith S Donath. Homophily in online dating: when do you like someone like yourself? In *CHI'05 Extended Abstracts on Human Factors in Computing Systems*, pages 1371–1374. ACM, 2005.

[25] Jacob G Foster, David V Foster, Peter Grassberger, and Maya Paczuski. Edge direction and the structure of networks. *Proceedings of the National Academy of Sciences*, 107(24):10815–10820, 2010.

[26] W. N. Francis and H. Kucera. Brown Corpus Manual. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US, 1979.

[27] L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of Relational Structure. *Journal of Machine Learning Research*, 3:679–707, 2002.

[28] Prem Gopalan, David Mimno, Sean Gerrish, Michael Freedman, and David Blei. Scalable inference of overlapping communities. In *Neural Information Processing Systems (NIPS)*, 2012.

[29] Amit Gruber, M Rosen-Zvi, and Y Weiss. Latent topic models for hypertext. In *Proc. 24th Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2008.

[30] Thomas Hofmann. Probabilistic latent semantic indexing. In *International ACM SIGIR conference on research and development in Information Retrieval (SIGIR)*, 1999.

[31] P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

[32] P. W. Holland and S. Leinhardt. An exponential family of probability distributions for directed graphs. *Journal of the American Statistical Association*, 76(373):33–50, 1981.

[33] Brian Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Phys. Rev. E*, 83:016107, 2011.

[34] Myunghwan Kim and Jure Leskovec. Latent Multi-group Membership Graph Model. *CoRR*, abs/1205.4546, 2012.

[35] F. Lorrain and H. C. White. Structural equivalence of individuals in social networks. *Journal of mathematical sociology*, 1(1):49–80, 1971.

[36] Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.

[37] Qing Lu and Lise Getoor. Link-based Classification. In *International Conference on Machine Learning (ICML)*, 2003.

[38] Qing Lu and Lise Getoor. Link-based Classification Using Labeled and Unlabeled Data. *ICML Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, 2003.

[39] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, pages 415–444, 2001.

[40] Marina Meilua. Comparing clusterings by the variation of information. *Learning theory and kernel machines*, pages 173–187, 2003.

[41] Cristopher Moore, Gourab Ghoshal, and M. E. J. Newman. Exact solutions for models of evolving networks with addition and deletion of nodes. *Phys. Rev. E*, 74:036121, 2006.

[42] Cristopher Moore, Xiaoran Yan, Yaojia Zhu, Jean-Baptiste Rouquier, and Terran Lane. Active learning for node classification in assortative and disassortative networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 841–849, 2011.

[43] M. Mørup and L. K. Hansen. Learning latent structure in complex networks. *NIPS Workshop on Analyzing Networks and Learning with Graphs*, 2009.

[44] Ramesh M. Nallapati, Amr Ahmed, Eric P. Xing, and William W. Cohen. Joint latent topic models for text and citations. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2008.

[45] M. E. J. Newman. Assortative Mixing in Networks. *Phys. Rev. Lett.*, 89:208701, 2002.

[46] M. E. J. Newman. Mixing patterns in networks. *Physical Review E*, 67(2):026126, 2003.

[47] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):036104, 2006.

[48] M. E. J. Newman and E. A. Leicht. Mixture models and exploratory analysis in networks. *Proc. National Academy of Sciences*, 104(23):9564–9, 2007.

[49] Gideon Schwarz. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[50] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, and Lise Getoor. Collective classification in network data. *AI Magazine*, pages 1–24, 2008.

[51] T. A. Snijders and K. Nowicki. Estimation and prediction for stochastic blockmodels for graphs with latent block structure. *Journal of Classification*, 14(1):75–100, 1997.

[52] Congkai Sun, Bin Gao, Zhenfu Cao, and Hang Li. HTM: A Topic Model for Hypertexts. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, 2008.

[53] S. Wasserman and C. Anderson. Stochastic a posteriori blockmodels: Construction and assessment. *Social Networks*, 9(1):1–36, 1987.

[54] H. C. White, S. A. Boorman, and R. L. Breiger. Social structure from multiple networks. I. Blockmodels of roles and positions. *American journal of sociology*, pages 730–780, 1976.

[55] Kevin S. Xu and Alfred O. Hero III. Dynamic stochastic blockmodels: Statistical models for time-evolving networks. *CoRR*, abs/1304.5974, 2013.

[56] Tianbao Yang, Rong Jin, Yun Chi, and Shenghuo Zhu. A Bayesian framework for community detection integrating content and link. In *Proc. Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2009.

[57] Philip Yu, Jiawei Han, and Christos Faloutsos. *Link Mining: Models, Algorithms, and Applications*. Springer, 2010.

[58] W. W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33(4):452–473, 1977.

[59] Yunpeng Zhao, Elizaveta Levina, and Ji Zhu. Link prediction for partially observed networks. *arXiv preprint arXiv:1301.7047*, 2013.

[60] Yaojia Zhu, Xiaoran Yan, Lise Getoor, and Cristopher Moore. Scalable text and link analysis with mixed-topic link models. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 473–481, 2013.

[61] Yaojia Zhu, Xiaoran Yan, and Cristopher Moore. Oriented and degree-generated block models: generating and inferring communities with inhomogeneous degree distributions . *Journal of Complex Networks*, 2013.