Stochastic Blockmodels meet Graph Neural Networks

Nikhil Mehta[*1] **Lawrence Carin**[1] **Piyush Rai**[2]

Abstract

Stochastic blockmodels (SBM) and their variants, $e.g.$, mixed-membership and overlapping stochastic blockmodels, are latent variable based generative models for graphs. They have proven to be successful for various tasks, such as discovering the community structure and link prediction on graph-structured data. Recently, graph neural networks, $e.g.$, graph convolutional networks, have also emerged as a promising approach to learn powerful representations (embeddings) for the nodes in the graph, by exploiting graph properties such as locality and invariance. In this work, we unify these two directions by developing a *sparse* variational autoencoder for graphs, that retains the interpretability of SBMs, while also enjoying the excellent predictive performance of graph neural nets. Moreover, our framework is accompanied by a fast recognition model that enables fast inference of the node embeddings (which are of independent interest for inference in SBM and its variants). Although we develop this framework for a particular type of SBM, namely the *overlapping* stochastic blockmodel, the proposed framework can be adapted readily for other types of SBMs. Experimental results on several benchmarks demonstrate encouraging results on link prediction while learning an interpretable latent structure that can be used for community discovery.

1. Introduction

Learning the latent structure in graph-structured data (Fortunato, 2010; Goldenberg et al., 2010; Schmidt & Morup, 2013) is an important problem in a wide range of domains, such as social and biological network analysis and recommender systems. These latent structures help discover the underlying communities in the network, as well as in predicting potential links between nodes. Latent space models (Hoff et al., 2002) and their structured extensions, such as the stochastic blockmodel (Nowicki & Snijders, 2001) and variants like the infinite relational model (IRM) (Kemp et al., 2006), mixed-membership stochastic blockmodel (MMSB) (Airoldi et al., 2008), and the overlapping stochastic blockmodel (OSBM) (Miller et al., 2009a; Latouche et al., 2011a) accomplish this by learning low-dimensional, interpretable node embeddings defined via structured latent variables. These embeddings can be used to identify the community membership(s) of each node in the graph, as well as for tasks such as link prediction.

The overlapping stochastic blockmodel (OSBM), also known as the latent feature relational model (LFRM), is a particularly appealing model for relational data (Miller et al., 2009a; Latouche et al., 2011a; Zhu et al., 2016). The OSBM/LFRM models each node in the graph as belonging to one or more communities using a binary membership vector, and defines the link probability between any pair of nodes as a *bilinear* function of their community membership vectors. Despite its appealing properties, the OSBM/LFRM has a number of limitations. In particular, although usually considered to be more expressive (Miller et al., 2009a) than models such as IRM and MMSB, a single layer of binary node embeddings and the bilinear model for the link generation can still limit the expressiveness of OSBM/LFRM. Moreover, it has a challenging inference procedure, which primarily relies on MCMC (Miller et al., 2009a; Latouche et al., 2011a) or mean-field variational inference (Zhu et al., 2016). Although recent models have tried to improve the expressiveness of OSBM/LFRM, $e.g.$, by assuming a *deep* hierarchy of binary-vector-based node embeddings (Hu et al., 2017), inference in such models remains intractable, requiring expensive MCMC-based inference. It is therefore desirable to have a model that retains the basic spirit to OSBM/LFRM ($e.g.$, easy interpretability and strong link prediction performance), but with greater expressiveness, and a simpler and scalable inference procedure.

Motivated by these desiderata, we develop a deep generative framework for graph-structured data, that inherits the easy interpretability of overlapping stochastic blockmodels, but is

[*]The major part of this work was done when Nikhil Mehta was at IIT Kanpur. [1]Department of Electrical and Computer Engineering, Duke University [2]Department of Computer Science, IIT Kanpur. Correspondence to: Nikhil Mehta <nikhilmehta.dce@gmail.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

much more expressive and enjoys a fast inference procedure. Our framework is based on a novel, *sparse* variant of the variational autoencoder (VAE) (Kingma & Welling, 2013), designed to model graph-structured data. Our VAE-based setup comprises a nonlinear generator/decoder for the graph and a nonlinear encoder based on the graph convolutional network (GCN) (Kipf & Welling, 2016a) (although other graph neural networks can also be used). Our framework posits each node of the graph to have an embedding in the form of a sparse latent representation (modeled by a Beta-Bernoulli process (Griffiths & Ghahramani, 2011), which also enables *learning* the size of the embeddings). The generator/decoder part of the VAE models the probability of a link between two nodes via a nonlinear function (defined by a deep neural network) of their associated embeddings. The encoder part of the VAE consists of a fast *recognition* model that is designed leveraging reparameterization method for Beta and Bernoulli distributions (Maddison et al., 2017; Nalisnick & Smyth, 2017). The recognition model, based on stochastic gradient variational Bayes (SGVB) inference, enables fast inference of the node embeddings. In contrast, the traditional stochastic blockmodels rely on iterative MCMC or variational inference procedures for inferring the node embeddings. Consequently, the SGVB inference algorithm we develop is also of independent interest, since the recognition model enables fast inference of the node embeddings in *single-layer* overlapping stochastic blockmodels.

2. Preliminaries

We first introduce notation and then briefly describe the overlapping stochastic blockmodel (OSBM) (Latouche et al., 2011a; Miller et al., 2009a; Zhu et al., 2016). As described in the next section, our deep generative model enriches OSBM by endowing it with a deep architecture based on a *sparse* variational autoencoder, and a fast inference algorithm based on a recognition model. We assume that the graph is given as an adjacency matrix $\mathbf{A} \in \{0,1\}^{N\times N}$, where N denotes the number of nodes. We assume $A_{nm} = 1$ if there exist a link between node n and node m, and otherwise $A_{nm} = 0$. In addition to $\mathbf{A}$, for each node we may also be provided node features. These are given in the form of an $N \times D$ matrix $\mathbf{X}$, with $\boldsymbol{x}_n \in \mathbb{R}^D$ being the node features for node n, and D being the number of observed features.

The OSBM (Latouche et al., 2011a; Miller et al., 2009a; Zhu et al., 2016) is a stochastic blockmodel for networks; it assumes each node n has an associated binary vector (node embedding), also termed a *latent feature vector* $\boldsymbol{z}_n \in \{0,1\}^K$. Within the node embedding, $z_{nk} = 1$ denotes that node n belongs to cluster/community k, and $z_{nk} = 0$ otherwise. The OSBM allows each node to simultaneously belong to multiple communities, and defines the link probability between two nodes via a bilinear function of their latent feature vectors

$$p(A_{nm} = 1|\boldsymbol{z}_n, \boldsymbol{z}_m, \mathbf{W}) = \sigma(\boldsymbol{z}_n^\top \mathbf{W} \boldsymbol{z}_m) \quad (1)$$

where entry $w_{k\ell}$ in $\mathbf{W} \in \mathbb{R}^{K\times K}$ affects the probability of a link between node n and node m belonging to cluster k and cluster ℓ, respectively.

The nonparametric latent feature relational model (LFRM) is a specific type of OSBM, that leverages the Indian Buffet Process (IBP) prior (Miller et al., 2009a) on the $N \times K$ binary matrix $\mathbf{Z} = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N]^\top$ of the node-community membership vectors. Use of the IBP enables *learning* the number of communities. Inference in LFRM/OSBM is typically performed via MCMC or variational inference (Miller et al., 2009a; Latouche et al., 2011a; Zhu et al., 2016), which tends to be slow and often cannot scale easily to more than a few hundred nodes.

3. Deep Generative OSBM

We now present our sparse VAE based deep generative framework for overlapping stochastic blockmodel. The proposed architecture, depicted in Fig. 1 (left), associates each link $A_{nm} \in \{0,1\}$ with two latent embeddings $\boldsymbol{z}_n$ and $\boldsymbol{z}_m$ (for the nodes associated with this link). Each link probability is modeled as a nonlinear function of the embeddings of its associated nodes. Unlike the standard VAE that assumes dense, Gaussian-distributed embeddings, since we wish to use the node embeddings to also infer the community membership(s) of each node (as it is one of the goals of stochastic blockmodels), we impose sparsity on the node embeddings. This is done by modeling them as a sparse vector of the form $\boldsymbol{z}_n = \boldsymbol{b}_n \odot \boldsymbol{r}_n$, where $\boldsymbol{b}_n \in \{0,1\}^K$ is a binary vector modeled using a stick-breaking process prior (Teh et al., 2007) and $\boldsymbol{r}_n \in \mathbb{R}^K$ is a real-valued vector with a Gaussian prior. Modeling $\boldsymbol{b}_n$ using the stick-breaking prior enables learning the node embedding size K from data. Note that, unlike the OSBM/LFRM, which assumes the node embedding $\boldsymbol{z}_n$ to be a strictly binary vector, our framework models it as a sparse real-valued vector, providing a more flexible and informative representation for the nodes. In particular, this enables inference of not just the node's membership into communities, but also the *strength* of the membership in each of the communities the node belongs to. Specifically, $b_{nk} \in \{0,1\}$ denotes whether node n belongs to cluster k or not, and $r_{nk} \in \mathbb{R}$ denotes the strength.

3.1. VAE Generator/Decoder

Given the node embeddings $\boldsymbol{z}_n = \boldsymbol{b}_n \odot \boldsymbol{r}_n$, the VAE decoder generates each link in the graph as $A_{nm} \sim p_\theta(\boldsymbol{z}_n, \boldsymbol{z}_m)$, where probability distribution p_θ defines a *decoder* or generator model for the graph. This decoder can consist of one or more layers of deterministic *nonlinear* transformation of the

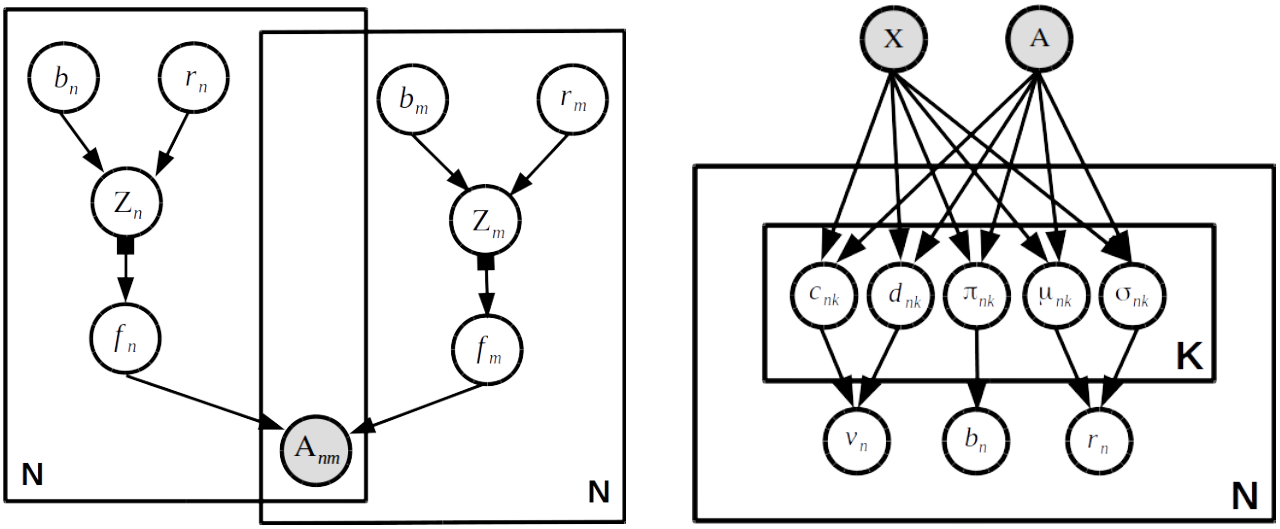


Figure 1. (Left) The generator/decoder model in plate notation. Note that the mapping from $\boldsymbol{z}_n$ to $\boldsymbol{f}_n$ is a deterministic nonlinear transformation, modeled by a deep neural network. (Right) The encoder model, defined by a graph convolutional network (Kipf & Welling, 2016a) that takes as input the network $\mathbf{A}$ and node features $\mathbf{X}$ (if available) and outputs the parameters of the variational distributions of the model parameters.

node embeddings $\boldsymbol{z}_n$. Denoting the overall transformation for a node embedding $\boldsymbol{z}_n$ as $f(\boldsymbol{z}_n) = \boldsymbol{f}_n$, we model the probability of a link as $p(A_{nm} = 1|\boldsymbol{f}_n, \boldsymbol{f}_m) = \sigma(\boldsymbol{f}_n^\top \boldsymbol{f}_m)$, where the nonlinear function f can be modeled by a deep neural network (in our experiments, we use a deep neural net with each hidden layer having leaky ReLU nonlinearity). Figure 1 (left) depicts the generator.

We model the binary vector $\boldsymbol{b}_n \in \{0, 1\}^K$, denoting node-community memberships, using the stick-breaking construction of the IBP (Teh et al., 2007), which enables learning of the *effective* K by specifying a sufficiently large truncation level K. The stick-breaking construction is given as follows

$$v_k \sim \text{Beta}(\alpha, 1), \quad k = 1, \ldots, K \tag{2}$$

$$\pi_k = \prod_{j=1}^{k} v_j, \quad b_{nk} \sim \text{Bernoulli}(\pi_k) \tag{3}$$

We further assume a Gaussian prior on membership strengths $\boldsymbol{r}_n \in \mathbb{R}^K$, i.e., $p(\boldsymbol{r}_n) = \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$.

3.2. VAE Encoder

We employ a *nonlinear* encoder to infer the node embedding $\boldsymbol{z}_n$ for each node, using a fast non-iterative *recognition model* (Kingma & Welling, 2013). Denoting the parameters of the variational posterior for the embeddings of all the nodes collectively as $\{\boldsymbol{v}, \boldsymbol{b}, \boldsymbol{r}\}$, we consider an approximation to the model's true posterior $p(\boldsymbol{v}, \boldsymbol{b}, \boldsymbol{r}|\mathbf{A}, \mathbf{X})$ with a variational posterior of the form $q_\phi(\boldsymbol{v}, \boldsymbol{b}, \boldsymbol{r})$. For simplicity, we consider a mean-field approximation, which allows us to factorize the posterior as $q_\phi(\boldsymbol{v}, \boldsymbol{b}, \boldsymbol{r}) = \prod_{k=1}^{K} \prod_{n=1}^{N} q_\phi(v_{nk}) q_\phi(b_{n,k}) q_\phi(r_{n,k})$. Our nonlinear encoder, as shown in Fig. 1 (Right), assumes variational distributions on the local variables of each node, *i.e.*, $\boldsymbol{v}_n$, $\boldsymbol{b}_n$ and $\boldsymbol{r}_n$, and defines the variational parameters of these distributions as the outputs of a graph convolutional network (GCN) (Kipf & Welling, 2016a), which takes as input the network $\mathbf{A}$ and the node feature matrix $\mathbf{X}$. GCN has recently emerged as a flexible encoder of graph-structured data (similar in spirit to convolutional neural networks for images), which makes it an ideal choice of the encoder in our VAE-based generative model for graphs. The forward propagation rule for each layer l in GCN is defined as $\mathbf{H}^l = g(\hat{\mathbf{A}}\mathbf{H}^{l-1}\mathbf{W}^l)$, where $\mathbf{H}^0 = \mathbf{X}$ ($\mathbf{X} = \mathbf{I}$ when no side information is present), $\mathbf{W}^l$ is the weight matrix, $g(\cdot)$ is the non-linear activation, and $\hat{\mathbf{A}}$ is the symmetric normalization of adjacency $\mathbf{A}$. Although here we have used the vanilla GCN in our architecture, more-generalized variants of GCN, such as GraphSAGE (Hamilton et al., 2017), can also be used as the encoder. The variational distributions have the following forms

$$q_\phi(v_{nk}) = \text{Beta}(c_{nk}, d_{nk}) \quad k = 1, \ldots, K \tag{4}$$

$$q_\phi(b_{nk}) = \text{Bernoulli}(\pi_{nk}) \quad k = 1, \ldots, K \tag{5}$$

$$q_\phi(\boldsymbol{r}_n) = \mathcal{N}(\boldsymbol{\mu}_n, \text{diag}(\boldsymbol{\sigma}_n^2)) \tag{6}$$

where $c_{nk}, d_{nk}, \pi_{nk}, \boldsymbol{\mu}_n$, and $\boldsymbol{\sigma}_n$ are outputs of a GCN, *i.e.*, $\{c_k, d_k, \pi_k, \mu_k, \sigma_k\}_{n=1}^{n=N} = \text{GCN}(\mathbf{A}, \mathbf{X})$. We use the stochastic gradient variational Bayes (SGVB) algorithm (Kingma & Welling, 2013) to infer the parameters of the variational distributions. Details on reparameterization and the loss formulation are provided in Section 4.

3.3. Special Cases

Existing models for graph-structured data can be seen as special cases of our framework. Recall that we model the node embeddings as $\boldsymbol{z}_n = \boldsymbol{b}_n \odot \boldsymbol{r}_n$, and our generative model is of the form $A_{nm} \sim p_\theta(\boldsymbol{z}_n, \boldsymbol{z}_m)$. If we ignore the community strength latent variable $\boldsymbol{r}_n$, *i.e.*, $\boldsymbol{z}_n$ is defined simply as $\boldsymbol{z}_n = \boldsymbol{b}_n$ (just a binary vector) and further define p_θ as a Bernoulli distribution with its probability being a bilinear function of the embeddings $\boldsymbol{z}_n$ and $\boldsymbol{z}_m$, then we recover the OSBM/LFRM (Latouche et al., 2011a; Miller et al., 2009a). Note, however, that while OSBM/LFRM typically rely on MCMC or variational inference, our framework can leverage SGVB for efficient inference.

Likewise, if we define $\boldsymbol{z}_n = \boldsymbol{r}_n$, *i.e.*, a *dense* vector, and define p_θ as a Bernoulli distribution with its probability being a bilinear function of the embeddings, we recover the Eigenmodel or latent-space model (LSM) (Hoff et al., 2002). Note that this model cannot infer K since the binary vector $\boldsymbol{b}_n$ is not present. Finally, if p_θ is a Bernoulli distribution with its probability being a *nonlinear* function of the embeddings, then we recover the VGAE model (Kipf & Welling, 2016b), which can also be seen as a nonlinear extension of LSM. Moreover, note that a key limitation of LSM and VGAE is that these cannot be used to infer the community structure (due to the non-sparse nature of $\boldsymbol{z}_n$) and usually can only be used for link-prediction tasks.

4. Inference

We define the factorized variational posterior $q_\phi(\boldsymbol{v}, \boldsymbol{b}, \boldsymbol{r})$ as

$$\begin{aligned} q_\phi(v_{nk}) &= \text{Beta}(v_{nk}|c_k(\mathbf{A},\mathbf{X}), d_k(\mathbf{A},\mathbf{X})) \\ q_\phi(b_{nk}) &= \text{Bernoulli}(b_{nk}|\pi_k(\mathbf{A},\mathbf{X})) \\ q_\phi(\boldsymbol{r}_n) &= \mathcal{N}(\boldsymbol{\mu}_n(\mathbf{A},\mathbf{X}), \text{diag}(\boldsymbol{\sigma}_n^2(\mathbf{A},\mathbf{X}))) \end{aligned}$$

where $c_k, d_k, \pi_k, \boldsymbol{\mu}_n$and $\boldsymbol{\sigma}_n$ are a function of the GCN encoder, with inputs $\mathbf{A}$ and $\mathbf{X}$. We define the loss function $\mathcal{L}$ parameterized by inference network (encoder) parameters (ϕ) and generator parameters (θ) by minimizing the *negative* of the evidence lower bound (ELBO)

$$\begin{aligned} \mathcal{L} = \sum_{n=1}^{N} \Big(&\text{KL}[q_\phi(\boldsymbol{b}_n|\boldsymbol{v}_n) \,||\, p_\theta(\boldsymbol{b}_n|\boldsymbol{v}_n)] + \text{KL}[q_\phi(\boldsymbol{r}_n) \,||\, p_\theta(\boldsymbol{r}_n)] \\ &+ \text{KL}[q_\phi(\boldsymbol{v}_n) \,||\, p(\boldsymbol{v}_n)] \Big) - \sum_{n=1}^{N} \Big(\mathbb{E}_q[\log p_\theta(X_n|\boldsymbol{z}_n)] \Big) \\ &- \sum_{n=1}^{N} \sum_{m=1}^{N} \Big(\mathbb{E}_q[\log p_\theta(A_{nm}|\boldsymbol{z}_n, \boldsymbol{z}_m)] \Big) \end{aligned} \tag{7}$$

where $\text{KL}[q(\cdot)||p(\cdot)]$ is the Kullback-Leibler divergence between $q(\cdot)$ and $p(\cdot)$. Note that here we have also included the loss from the reconstruction of the side information X_n. We have considered that the side information $\mathbf{X}$ and the links $\mathbf{A}$ are conditionally independent given the node embeddings $\boldsymbol{z}_{1:N}$. When there is no side information, we can ignore the $\mathbb{E}_q[\log p_\theta(X_n|\boldsymbol{z}_n)]$ term in the loss function. For the encoder and decoder parameters we infer point estimates, while we learn the distribution over the latent variables $\boldsymbol{b}, \boldsymbol{v}$, and $\boldsymbol{r}$.

Our variational autoencoder for link generation is trained using Stochastic Gradient Variational Bayes (SGVB) (Kingma & Welling, 2013). SGVB can be used to perform inference for a broad class of non-conjugate models and is therefore appealing to Bayesian nonparametric models, such as those based on stick-breaking priors that we use in our framework. SGVB uses differentiable Monte Carlo (MC) expectations to learn the model parameters. Specifically, it requires *differentiable, non-centered parameterization* (DNCP) (Kingma & Welling, 2014) to allow backpropagation. However, our model has expectations over Beta and Bernoulli distributions, neither of which permit easy reparameterization as required by SGVB. We leverage recent developments on reparameterizing these distributions (Maddison et al., 2017; Nalisnick & Smyth, 2017), which consequently leads to a simple inference procedure.

Following (Nalisnick & Smyth, 2017), we approximate the Beta posterior in (4) with the Kumaraswamy distribution, defined as: $Kumar(x; a, b) = abx^{a-1}(1-x^a)^{b-1}$ for $x \in (0, 1)$ and $a, b > 0$. The closed-form inverse CDF allows easy reparameterization, and samples for v_{nk} (with parameters c_{nk} and d_{nk}) can be drawn using:

$$\begin{aligned} u &\sim \text{Uniform}(0, 1) \\ v_{nk} &\overset{d}{=} (1 - u^{\frac{1}{d_{nk}}})^{\frac{1}{c_{nk}}} \end{aligned} \tag{8}$$

We compute the KL divergence between the Kumaraswamy $q(\boldsymbol{v})$ and the Beta distribution $p(\boldsymbol{v})$ by taking a finite approximation of the infinite sum as mentioned in (Nalisnick & Smyth, 2017).

For the Bernoulli random variable, we use the Binary Concrete distribution (Maddison et al., 2017; Jang et al., 2017) at the time of training, as a continuous relaxation to get the biased low-variance estimates of the gradient. The KL divergence between two Bernoulli distributions is relaxed using two Binary Concrete distributions.

We reparameterize b_{nk}, defined by a Bernoulli with probability π_{nk}, (in (3) and (5)) with reparameterization:

$$\begin{aligned} L &= \log\left(\frac{u}{1-u}\right) \\ b_{nk} &\overset{d}{=} \sigma\left(\frac{\text{logit}(\pi_{nk}) + L}{\lambda}\right) \end{aligned} \tag{9}$$

where $\sigma(\cdot)$ is the sigmoid function, $logit(\cdot)$ is the inverse-sigmoid function, λ is the relaxation temperature and $u \sim \text{Uniform}(0, 1)$.

Structured Mean-Field: Since the vanilla mean-field variational inference ignores the posterior dependence among the latent variables, we also considered Structured Stochastic Variational Inference (SSVI) (Hoffman, 2014; Hoffman et al., 2013), which allows global-local parameter dependency and improves upon the mean-field approximation. We considered $\boldsymbol{v}$ (and its variational parameters c and d) as global parameters and impose a hierarchical structure on $\boldsymbol{b}_n$ by conditioning it on $\boldsymbol{v}$. The variational posterior of our framework using SSVI can be factorized as $q_\phi(\boldsymbol{v}, \boldsymbol{b}, \boldsymbol{r}) = \prod_{k=1}^{K} q_\phi(v_k) \prod_{n=1}^{N} q_\phi(b_{n,k}|\boldsymbol{v}) q_\phi(r_{n,k})$ with $q_\phi(v_k) = \text{Beta}(c_k, d_k)$; $q_\phi(b_{nk}|\boldsymbol{v}) = \text{Bernoulli}(\pi_k)$; $\pi_k = \prod_{j=1}^{K} v_k$, where c_k, d_k are parameters to be learned. In practice, we have found structured mean-field to perform better than the mean-field, and our model implementation uses the former.

5. Related Work

The proposed framework can be seen as bridging two lines of research on modeling graphs: (i) structured latent variable models for graphs, such as stochastic blockmodels and its variants (Kemp et al., 2006; Airoldi et al., 2008; Miller et al., 2009a; Latouche et al., 2011a); and (ii) deep learning models for graphs, such as graph convolutional networks (Kipf & Welling, 2016a). Our effort is motivated by the goal of harnessing their complementary strengths to develop a deep generative stochastic blockmodel for graphs, that also enjoys an efficient inference procedure.

The most prominent methods in stochastic blockmodels include models that associate each node to a single community (Nowicki & Snijders, 2001; Kemp et al., 2006), a mixture of communities (Airoldi et al., 2008), and an overlapping set of communities (Miller et al., 2009a; Latouche et al., 2011a; Yang & Leskovec, 2012; Zhou, 2015). While stochastic blockmodels have nice interpretability, these models usually assume the links of the networks to be modeled as a simple bilinear function of the node embeddings, which may not be able to capture the nonlinear interactions between the nodes (Yan et al., 2011). An approach to model such nonlinear interactions was proposed in (Yan et al., 2011), using a matrix-variate Gaussian process. However, despite the modeling flexibility, inference in this model is challenging and the model is usually infeasible to run on networks with more than 100 nodes.

There is also significant recent interest in non-probabilistic deep learning models for graphs. Some of the prominent works in this direction include DeepWalk (Perozzi et al., 2014) and graph autoencoders (GAE) (Kipf & Welling, 2016a; Hamilton et al., 2017). DeepWalk is inspired by the idea of word embeddings. It treats each node as a "document," by starting a random walk at that node and taking the nodes encountered in the path taken as the word in that document. It uses document/word embedding methods to the learn embedding of each node. In contrast, the GAE approaches are based on the idea of graph convolutional networks (GCN) (Kipf & Welling, 2016a). This line of work nicely complements our contribution, since modules like GCN can be effectively used to design the encoder model for our deep generative framework. In particular, as noted in the model description, our encoder is essentially a GCN. We believe that such advances in graph encoding can be used as modules to design new deep generative models for relational data.

Despite the significant success of deep generative models for images and text data, there has been relatively little work on deep generative models for relational data (You et al., 2018; Hu et al., 2017; Wang et al., 2017; Kipf & Welling, 2016b). GraphRNN (You et al., 2018) learns a single representation of an entire graph to model the joint distribution of different graphs. The focus of GraphRNN is on generating small-sized graphs, whereas we focus on link prediction and community detection for a given graph. Among other existing methods, (Hu et al., 2017) proposed an extension of the LFRM via a deep hierarchy of binary latent features for each node. However, this model relies on expensive batch MCMC inference, precluding its applicability to large-scale networks. Another deep latent variable model was proposed recently in (Wang et al., 2017). However, this model also has a difficult inference procedure, requiring model-specific inference. Moreover, the node embeddings are not interpretable. Perhaps the closest in spirit to our work is the recent work on variational graph autoencoders (VGAE) (Kipf & Welling, 2016b). Graphite (Grover et al., 2018) extends the VGAE by using a multi-layer iterative decoder that alternates between message passing and graph refinement. A similar decoding scheme can also be applied in our framework; however, the focus of this work is on learning sparse interpretable node embeddings. Both VGAE and Graphite are built on top of the standard VAE, and consequently do not have direct interpretability of node embeddings as desired by stochastic blockmodels. This leads to a model with different properties and a different inference procedure, compared to (Kipf & Welling, 2016b). Moreover, our VAE architecture is nonparametric in nature and can infer the node embedding size.

6. Experiments

We report experimental results on several synthetic and real-world datasets, to demonstrate the efficacy of our model. Our experimental results include quantitative comparisons on the task of link prediction as well as qualitative results, such as using the embeddings to discover the underlying communities in the network data. The qualitative results are meant to demonstrate the expressiveness of the latent space that our model infers. The expressive nature of our model

is the result of the sparse and interpretable embedding for each node of the graph. In particular, we show that these sparse embeddings can be interpreted as the memberships and strength of memberships of each node in one or more communities.

First we evaluate our model on link-prediction, comparing it with various baselines on several benchmark datasets on moderate (about 2000 nodes) to large-scale (about 20,000 nodes) datasets. We then analyze the latent structure $\boldsymbol{z}_n$ learned by our model on a synthetic and a real-world co-authorship dataset. We compare the latent structure with the embeddings learned by the variational graph autoencoder (VGAE) (Kipf & Welling, 2016b). We also examine the community structure on the real-world co-authorship dataset, and show that the proposed framework is able to readily capture the underlying communities. We refer to our framework as DGLFRM (Deep Generative Latent Feature Relational Model), which refers to our most general model with sparse embeddings $\boldsymbol{z}_n = \boldsymbol{b}_n \odot \boldsymbol{r}_n$ with nonlinear generator and nonlinear encoder. We also consider a variant of DGLFRM with binary embeddings $\boldsymbol{z}_n = \boldsymbol{b}_n$, which we refer to as DGLFRM-B (the 'B' here denotes "binary"). Note that DGLFRM-B can be seen as a deep generalization of LFRM (Miller et al., 2009b)/OSBM (Latouche et al., 2011b), with another key difference from LFRM/OSBM being the fact that we use amortized inference.

6.1. Baselines

For link prediction, we compare the proposed model with four baselines, one of which is a simplified variant of DGLFRM akin to LFRM (Miller et al., 2009a), which is an overlapping stochastic blockmodel. The original LFRM, which uses MCMC-based inference, was infeasible to run on the datasets used in these experiments. On the other hand, DGLFRM with $\boldsymbol{z}_n = \boldsymbol{b}_n$ and bilinear decoder (link generation model) is similar to LFRM, but with a much faster SGVB based inference (we will refer to this simplified variant of DGLFRM as LFRM).

Among the other three baselines, Spectral Clustering (SC) and DeepWalk (DW) (Perozzi et al., 2014) learn node embeddings, which we use to compute the link probability as $\sigma(\boldsymbol{z}_n^\top \boldsymbol{z}_m)$. The third baseline is the recently proposed variational autoencoder on graphs (VGAE) (Kipf & Welling, 2016b). Note that none of these baselines can be used for community detection, since the real-valued embeddings learned by these baselines are not interpretable (unlike our model which learns sparse embeddings, with nonzeros denoting community memberships).

6.2. Datasets

We consider five real-world datasets, with three datasets consisting of side information in the form of the node features, and the other two datasets having only the link information. For the link-prediction experiments, all models are provided a partially-complete network (with unknown part to be predicted). The node features (when available) are provided to all the models. The description of each data set is as follows:

- **NIPS12**: The NIPS12 coauthor network (Zhou, 2015) includes all 2037 authors in NIPS papers from volumes 1-12, with 3134 edges. This network has no side information.
- **Yeast**: The Yeast protein interaction network (Zhou, 2015) has 2361 nodes and 6646 non-self edges. This network has no side information.
- **Cora**: Cora network is a citation network consisting of 2708 documents. The datasets contain sparse bag-of-words feature vectors of length 1433 for each document. These are used as node features. The network has total 5278 links.
- **Citeseer**: Citeseer is a citation network consisting of 3312 scientific publications from six categories: agents, AI, databases, human computer interaction, machine learning, and information retrieval. The side information for the dataset is the category label for each paper which is converted into a one-hot representation. These one-hot vectors are used as node features. The network has a total of 4552 links.
- **Pubmed**: A citation network consisting of 19,717 nodes. The dataset contains sparse bag-of-words feature vectors of length 500 for each document, used as node features. The network has total 44,324 links.

6.3. Link Prediction

We use Area Under the ROC Curve (AUC) and Average Precision (AP) to compare our model with the other baselines for link prediction. For all datasets, we hold out 10% and 5% of the links as our test set and validation set, respectively, and use the validation set to fine-tune the hyperparameters. We take the average of AUC-ROC and AP scores by running our model on 10 random splits of each dataset, to compare with the baselines. The AUC-ROC scores of our models and the various baselines are shown in Table 1 and AP scores are shown in Table 2. As shown in the tables, our models outperforms the baselines on almost all datasets. We again highlight that unlike the baselines, such as VGAE, that cannot learn interpretable embeddings, our model also learns embeddings that can be interpreted as memberships of nodes into communities. The superior results of DGLFRM and DGLRFM-B demonstrate the benefit of our deep generative models. The significantly better results of these as compared to LFRM also show the benefit of endowing

Table 1. AUC ROC

Method	NIPS12	Yeast	Cora	Citeseer	Pubmed
SC	$0.8792 \pm .0003$	$0.7886 \pm .0001$	$0.8460 \pm .0001$	$0.8050 \pm .0001$	$0.8420 \pm .0002$
DW	$0.8058 \pm .0000$	$0.6443 \pm .0003$	$0.8310 \pm .0001$	$0.8050 \pm .0002$	$0.8440 \pm .0000$
VGAE	$0.8790 \pm .0055$	$0.7784 \pm .0002$	$0.9260 \pm .0001$	$0.9080 \pm .0002$	$\mathbf{0.9418} \pm .0076$
LFRM	$0.8489 \pm .0001$	$0.7975 \pm .0006$	$0.9096 \pm .0026$	$0.8965 \pm .0035$	$0.9152 \pm .0041$
DGLFRM-B	$\mathbf{0.8898} \pm .0028$	$\mathbf{0.8061} \pm .0003$	$0.9281 \pm .0024$	$0.9007 \pm .0020$	$0.9396 \pm .0052$
DGLFRM	$0.8734 \pm .0043$	$0.7856 \pm .0005$	$\mathbf{0.9343} \pm .0023$	$\mathbf{0.9379} \pm .0032$	$0.9395 \pm .0008$

Table 2. Average Precision (AP).

Method	NIPS12	Yeast	Cora	Citeseer	Pubmed
SC	$0.9022 \pm .0002$	$0.8440 \pm .0001$	$0.8850 \pm .0000$	$0.8500 \pm .0100$	$0.8780 \pm .0100$
DW	$0.8634 \pm .0000$	$0.6699 \pm .0002$	$0.8500 \pm .0001$	$0.8360 \pm .0001$	$0.8440 \pm .0000$
VGAE	$0.9114 \pm .0042$	$0.8349 \pm .0002$	$0.9328 \pm .0001$	$0.9200 \pm .0002$	$0.9394 \pm .0088$
LFRM	$0.8870 \pm .0000$	$0.8268 \pm .0005$	$0.9060 \pm .0033$	$0.9118 \pm .0031$	$0.9197 \pm .0054$
DGLFRM-B	$\mathbf{0.9120} \pm .0021$	$\mathbf{0.8442} \pm .0002$	$0.9259 \pm .0023$	$0.9153 \pm .0031$	$0.9454 \pm .0050$
DGLFRM	$0.9005 \pm .0027$	$0.8388 \pm .0002$	$\mathbf{0.9376} \pm .0022$	$\mathbf{0.9438} \pm .0073$	$\mathbf{0.9497} \pm .0035$

Table 3. Example of communities inferred by our model on the NIPS data.

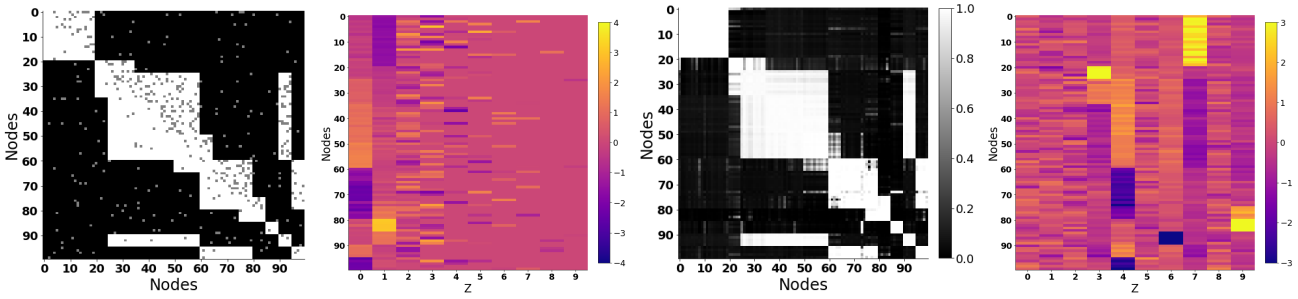
Cluster	Authors
Probabilistic Modeling	Sejnowski T, Hinton G, Dayan P, Jordan M, Williams C
Reinforcement Learning	Barto A, Singh S, Sutton R, Connolly C, Precup D
Robotics/Vision	Shibata T, Peper F, Thrun S, Giles C, Michel A
Computational Neuroscience	Baldi P, Stein C, Rinott Y, Weinshall D, Druzinsky R
Neural Networks	Pearlmutter B Abu-Mostafa Y, LeCun Y, Sejnowski T, Tang A

LFRM with a deep architecture, with nonlinear decoder and nonlinear encoder. The hyperparameter settings used for all experiments are included in the Supplementary Material. We also performed an experiment to investigate the model's ability to leverage node features. As expected, when using the features the model performs better compared to the case when it ignores features. This experiment is included in the supplementary section.

6.4. Qualitative Analysis on Learned Embeddings

To demonstrate the interpretable nature of the embeddings learned by our model, we generate a synthetic dataset with 100 nodes and 10 communities. The dataset is generated by fixing the ground-truth communities (by creating a binary vector for each node) such that some of the nodes belong to same communities. The adjacency matrix is then generated using a simple inner product, followed by the sigmoid operation (Figure 2a). We train using 85% of the synthetic adjacency matrix for link-prediction and for visualizing the latent structure that our model learns. The latent structure obtained using DGLFRM is plotted in Figure 2(b). Figure 2(c) shows that by using only the first two dimensions of the latent structure we can reconstruct the graph reasonably well. This depicts an important property of using a stick-breaking IBP prior which encourages the most commonly selected communities (the columns on the left in Figure 2 (b)) to be dense, while the communities with higher indices (columns in right) to be sparse. This shows that DGLFRM can learn the effective number of communities given a graph. Finally, we can quantize the latent space into discrete intervals to extract nodes belonging to different communities. In our experiments we saw that the latent structure learned is in fact close to the ground-truth community assignments we started with. In Figure 2(d) we compare the community structure from our model with the latent structure obtained by running the VGAE. Note that the Gaussian latent structure in VGAE is dense and therefore fails to learn community memberships that are readily interpretable.

We also do a qualitative analysis on the NIPS12 dataset. Again we train DGLFRM and VGAE using 85% of the adjacency matrix. Table 3 shows five of the inferred communities by DGLFRM. The authors shown under each community are ordered by the strength of their community memberships (in decreasing order). As Table 3 shows, each of the communities represent a sub-field, with authors working on similar topics. Moreover, note that some authors ($e.g.$,



(a) Adjacency (b) DGLFRM (c) Generated (d) VGAE

Figure 2. (a) The synthetic adjacency matrix (where white, black and grey denote link, no-link and hidden parts of the graph). (b) The latent structure of synthetic data learned using DGLFRM. The truncation parameter (K) of the latent structure was fixed to 10. (c) The graph generated by DGLFRM using only the first 2 dimensions of the latent structure($i.e.$ columns 0 and 1 in (b)). The columns 2-9 were all set to zero. The graph is represented as the probability of links; white (black) represents link (no-link) with high probability. (d) The latent structure of synthetic data learned using VGAE.

Sejnowski) are inferred as belonging to more than one community. This qualitative experiment demonstrates that our model can learn interpretable embeddings that can be used for tasks such as (overlapping) clustering. We have included a visualization of the latent structure learned on NIPS12 data in the Supplementary Material. Note that our model can infer the number of communities naturally, via the stick-breaking prior. The stick-breaking prior requires specifying a large truncation level on the number of communities. Our model effectively infers the "active" communities for a given truncation level. As shown in Fig. 2 (b)-(c), the posterior inference in our model is able to "turn off" the unnecessary columns in $\mathbf{Z}$. Although we do not know the ground truth for the number of communities, the number of inferred active communities is similar to what is reported in prior work on nonparametric Bayesian overlapping stochastic blockmodels (Miller et al., 2009a). Note that VGAE embeddings require an additional step (such as K-Means clustering) to cluster nodes. Moreover, a method such as K-means cannot detect overlapping communities, and it is also sensitive to the initialization of K (estimated number of communities). For reference, we have included the clustering results on the VGAE embeddings in the Supplementary Material.

7. Conclusion and Discussion

We have presented a deep generative framework for overlapping community discovery and link prediction. This work combines the interpretability of stochastic blockmodels, such as the latent feature relational model, with the modeling power of deep generative models. Moreover, leveraging a nonparametric Bayesian prior on the node embeddings enables learning the node embedding size ($i.e.$, the number of communities) from data. Our framework is modular and a wide variety of decoder and encoder models can be used. In particular, it can leverage recent advances in non-probabilistic autoencoders for graphs, such as the graph convolutional network (Kipf & Welling, 2016a) or its extensions (Hamilton et al., 2017). Inference in the model is based on SGVB, that does not require conjugacy. This further widens the applicability of our framework to model different types of networks ($e.g.$, weighted, count-valued edges, and power-law degree distribution of node degrees). We believe this combination of discrete latent variables based stochastic blockmodels and graph neural network will help leverage their respective strengths, and will fuel further research and advance the state-of-the-art in (deep) generative modeling of graph-structured data.

Although SGVB inference makes our model fairly efficient, it can be scaled up further for massive networks by using mini-batch based inference (Chen et al., 2018). Another possibility to scale up the model is to replace the Bernoulli-logistic likelihood model by a Bernoulli-Poisson link (Zhou, 2015), which enable scaling up the model in the number of nonzeros ($i.e.$, number of edges) in the network. Given that our framework can work with a wide variety of decoder/generator models, such modifications can be done without much difficulty.

Finally, in this work we model each node as having a single binary vector, denoting its memberships in one or more communities. Another interesting extension would be to consider multiple layers of latent variables, which can model a node's membership into a hierarchy of communities (Ho et al., 2011; Blundell & Teh, 2013; Hu et al., 2017).

Acknowledgements: PR acknowledges support from Google AI/ML faculty award and DST-SERB Early Career Research Award. The Duke investigators acknowledge the support of DARPA and ONR.

References

Airoldi, E. M., Blei, D. M., Fienberg, S. E., and Xing, E. P. Mixed membership stochastic blockmodels. *JMLR*, 2008.

Blundell, C. and Teh, Y. W. Bayesian hierarchical community discovery. In *NIPS*, 2013.

Chen, J., Ma, T., and Xiao, C. Fastgcn: Fast learning with graph convolutional networks via importance sampling. *CoRR*, abs/1801.10247, 2018. URL http://arxiv.org/abs/1801.10247.

Fortunato, S. Community detection in graphs. *Physics reports*, 486(3):75–174, 2010.

Goldenberg, A., Zheng, A. X., Fienberg, S. E., and Airoldi, E. M. A survey of statistical network models. *Foundations and Trends® in Machine Learning*, 2010.

Griffiths, T. L. and Ghahramani, Z. The indian buffet process: An introduction and review. *JMLR*, 2011.

Grover, A., Zweig, A., and Ermon, S. Graphite: Iterative generative modeling of graphs. *arXiv preprint arXiv:1803.10459*, 2018.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. In *NIPS*, 2017.

Ho, Q., Parikh, A., Song, L., and Xing, E. Multiscale community blockmodel for network exploration. In *AISTATS*, 2011.

Hoff, P. D., Raftery, A. E., and Handcock, M. S. Latent space approaches to social network analysis. *JASA*, 2002.

Hoffman, M. D. Stochastic structured mean-field variational inference. *CoRR*, abs/1404.4114, 2014.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Hu, C., Rai, P., and Carin, L. Deep generative models for relational data with side information. In *International Conference on Machine Learning*, pp. 1578–1586, 2017.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017.

Kemp, C., Tenenbaum, J. B., Griffiths, T. L., Yamada, T., and Ueda, N. Learning systems of concepts with an infinite relational model. In *Proceedings of the national conference on artificial intelligence*, 2006.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kingma, D. P. and Welling, M. Efficient gradient-based inference through transformations between bayes nets and neural nets. *CoRR*, abs/1402.0480, 2014. URL http://arxiv.org/abs/1402.0480.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016a.

Kipf, T. N. and Welling, M. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016b.

Latouche, P., Birmelé, E., and Ambroise, C. Overlapping stochastic block models with application to the french political blogosphere. *The Annals of Applied Statistics*, 2011a.

Latouche, P., Birmel, E., and Ambroise, C. Overlapping stochastic block models with application to the french political blogosphere. *The Annals of Applied Statistics*, 2011b.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. In *ICLR*, 2017.

Miller, K., Griffiths, T., and Jordan, M. Nonparametric latent feature models for link prediction. *NIPS*, 2009a.

Miller, K. T., Griffiths, T. L., and Jordan, M. I. Nonparametric latent feature models for link prediction. In *NIPS*, 2009b.

Nalisnick, E. and Smyth, P. Stick-breaking variational autoencoders. In *ICLR*, 2017.

Nowicki, K. and Snijders, T. A. B. Estimation and prediction for stochastic blockstructures. *JASA*, 2001.

Perozzi, B., Al-Rfou, R., and Skiena, S. Deepwalk: Online learning of social representations. In *KDD*, 2014.

Schmidt, M. N. and Morup, M. Nonparametric bayesian modeling of complex networks: An introduction. *Signal Processing Magazine, IEEE*, 30(3), 2013.

Teh, Y. W., Grr, D., and Ghahramani, Z. Stick-breaking construction for the indian buffet process. In *AISTATS*, 2007.

Wang, H., Shi, X., and Yeung, D.-Y. Relational deep learning: A deep latent variable model for link prediction. In *AAAI*, pp. 2688–2694, 2017.

Yan, F., Xu, Z., and Qi, Y. Sparse matrix-variate gaussian process blockmodels for network modeling. In *UAI*, 2011.

Yang, J. and Leskovec, J. Community-affiliation graph model for overlapping network community detection. In *ICDM*, 2012.

You, J., Ying, R., Ren, X., Hamilton, W. L., and Leskovec, J. Graphrnn: A deep generative model for graphs. *CoRR*, abs/1802.08773, 2018. URL http://arxiv.org/abs/1802.08773.

Zhou, M. Infinite edge partition models for overlapping community detection and link prediction. In *AISTATS*, 2015.

Zhu, J., Song, J., and Chen, B. Max-margin nonparametric latent feature models for link prediction. *arXiv preprint arXiv:1602.07428*, 2016.