A DEEP LATENT SPACE MODEL FOR DIRECTED GRAPH REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph representation learning is a fundamental problem for modeling relational data and benefits a number of downstream applications. Traditional Bayesian-based random graph models and recent deep learning based methods are complementary to each other in interpretability and scalability. To take the advantages of both models, some combined methods have been proposed. However, existing models are mainly designed for *undirected graphs*, while a large portion of real-world graphs are directed. The focus of this paper is on the more challenging *directed graphs* where both the existences and directions of edges need to be learned. We propose a Deep Latent Space Model (DLSM) for directed graphs to incorporate the traditional latent space random graph model into deep learning frameworks via a hierarchical variational auto-encoder architecture. To adapt to directed graphs, our model generates multiple highly interpretable latent variables as node representations, and the interpretability of representing node influences is theoretically proved. The experimental results on real-world graphs demonstrate that our proposed model achieves the state-of-the-art performances on link prediction and community detection tasks while generating interpretable node representations of directed graphs.

1 INTRODUCTION

Learning representations on graphs is a fundamental problem for graph analysis and benefits a number of downstream applications, typically including link prediction, community detection and influential node identification. Traditionally, a plethora of Bayesian-based random graph models have been proposed for learning graph representations (Holland et al., 1983; Hoff et al., 2002; Karrer & Newman, 2011; Sewell & Chen, 2015). Despite the ideal interpretability of these methods, they are unscalable to model large-scale graphs due to the expensive iterative inference procedures. Boosted by the powerful representation learning ability, graph neural networks (GNNs) have been proposed to learn the topology of graph-structured data (Kipf & Welling, 2017; Hamilton et al., 2017; Veličković et al., 2018; Zeng et al., 2020). However, these GNN-based methods usually bring about the interpretability issues, which restrict the practical applications of these methods.

As an effective deep learning method based on the Bayesian theory, the variational auto-encoders (VAEs) (Kingma & Welling, 2014b) have been proposed for representation learning on graph-structured data (Kipf & Welling, 2016; Grover et al., 2019; Salha et al., 2019). To capture the mutual facilitation of random graph models and deep learning based methods, some recent research attempts to combine them using VAEs, and build the deep generative random graph models (Mehta et al., 2019; Sarkar et al., 2020). However, most of these VAE-based methods are designed only for *undirected graphs*. In contrast, a large portion of real-world graphs, such as social and citation networks, are *directed graphs*, of which undirected graphs are special cases (i.e. an undirected edge can be regarded as a bidirectional edge). The representation learning on directed graphs is particularly challenging since not only the existences but also the directions of edges need to be learned. Although directed graphs have been extensively studied by traditional research on random graph models (Hoff et al., 2002; Krivitsky et al., 2009; Sewell & Chen, 2015), there are still few deep learning based methods and no combined models existed to address them.

In this paper, we present the first VAE-based representation learning model for directed graphs and propose the Deep Latent Space Model (DLSM), which combines graph convolutional networks

(GCNs) with the Bayesian random graph model using a hierarchical VAE architecture. Within our model, a deep GCN is leveraged to encode the asymmetric adjacency matrix and node features as hidden states, which are layer-wise transformed to variational parameters to generate latent random variables as interpretable node representations. Then, the hierarchical decoder network reconstructs the adjacency matrix using the learned representations.

To better capture the characteristics of graphs for deep random graph models, our method generates three types of interpretable representations, including the latent positions, node random factors and community memberships. The latent positions measure the distances between nodes and enable our method to model the *link reciprocity* of directed graphs, which means that both bidirectional and unidirectional edges exist. To characterize the *degree heterogeneity*, we introduce a pair of node random factors, namely the social activity and popularity, thus the asymmetric structure of directed edges can be modeled. Besides, we use a binary latent variable, named the community membership, to accommodate an overlapping *community structure*. We further prove that these node-specific variables, including the latent positions and random factors, are interpretable for representing the existence and strength of node influences.

For fast inference, we leveraged the stochastic gradient variational Bayes (SGVB) algorithm (Kingma & Welling, 2014a), which is far more efficient and scalable compared to the traditional iterative inference methods. We conduct experiments on link prediction task to verify the effectiveness of directed graph modeling with several real-world graphs, and further demonstrate the interpretability of latent variables on community detection task.

The main contributions of this work are summarized as follows:

1. We propose the DLSM model for directed graphs to incorporate the latent space model into deep learning frameworks using a hierarchical VAE architecture, which can generate highly interpretable representations of graph nodes for multiple downstream tasks.
2. To adapt our method to directed graphs, we introduce the social activity and popularity factors of each node, with theoretical proof of their capabilities in representing node influences.
3. We conduct experiments on five real-world graph datasets, and the results show that our proposed method can well support downstream tasks and achieve the state-of-the-art performances in link prediction and community detection.

2 Related Work

In this section, we briefly review the traditional Bayesian-based random graph models and deep learning based generative graph representation methods.

2.1 Bayesian-based random graph models

Classic Bayesian-based random graph models have developed for decades and are still valued in modeling relational data. This methods regard graph nodes as random variables following a prior distribution, and learns low-dimensional embeddings via the posterior distributions of nodes. One of the most well-known methods is the stochastic blockmodel (SBM) (Holland et al., 1983), which generates a latent variable indicating the community membership of each node. Following SBM, a large number of variants have been proposed, such as allowing an overlapping community structure (Miller et al., 2009) and involving degree heterogeneity (Karrer & Newman, 2011).

Another stream of research is the latent space model (LSM) (Hoff et al., 2002), which assumes each node as a position in an unobservable latent space and use distances to measure the relationships between nodes. Such method has soon been extended to model directed graphs by involving the degree heterogeneity of nodes (Krivitsky et al., 2009; Sewell & Chen, 2015). One of the most important advantages for LSM is that the symmetry of distances can naturally endow node embeddings with the link reciprocity of directed graphs. Although these above methods provide good theoretical properties and learn interpretable node embeddings, they rely on either the MCMC posterior sampling or mean-field variational inference with dramatically high computational complexity, and thus are usually unfeasible when modeling large-scale graphs.

2.2 DEEP LEARNING BASED GRAPH MODELS

The intriguing achievements of deep learning models for learning representations of Euclidean data have encouraged efforts to employ them on graph-structured data. The earliest attempts include DeepWalk (Perozzi et al., 2014) and node2vec (Grover & Leskovec, 2016), both of which encode local relations as node representations by conducting random walks on a graph. More recently, many GNN-based deep generative models have been proposed. Kipf & Welling (2017) first leverages the spectral GCN to generate node embeddings using the global topology of a graph. GraphSAGE (Hamilton et al., 2017) samples a fixed-size neighborhood of each node for the inductive representation learning, which allows unseen nodes to be excluded during training. Later, GraphGAN (Wang et al., 2018) unifies a generator and a discriminator to learn node embeddings by playing a minimax game. The graph attention networks (Veličković et al., 2018) introduce the attention mechanism to aggregate the information from neighbors. Enlightened by the powerful ability of VAEs for representation learning Kingma & Welling (2014b); Sønderby et al. (2016), the variational graph auto-encoders (VGAE) (Kipf & Welling, 2016) is proposed to generate Normal node embeddings using the VAE (Kingma & Welling, 2014b), which is then modified by Graphite (Grover et al., 2019) with a perception component based on GCN. Salha et al. (2019) proposes an asymmetric decoding scheme which handles unidirectional graph proximity.

To take advantages of the nice theoretical properties of random graph models and the powerful representation ability of deep learning methods, some research work tries to combine these two approaches for graph representation learning. Mehta et al. (2019) first combines the classic SBM with GCN to learn a sparse real-valued node embedding interpreted as the membership and strengths of each node belonging to different communities. Furthermore, Sarkar et al. (2020) builds a sparse ladder VAE (Sønderby et al., 2016) architecture to discover the communities at multiple levels of granularities. All of these combined models, however, are only designed for undirected graphs without considering graph characteristics, such as degree heterogeneity. In this paper, we design a hierarchical VAE architecture which combines the LSM and GCN to learn representations on directed graphs involving the community structure and degree heterogeneity.

3 DEEP LATENT SPACE MODEL

Consider a directed graph containing n nodes. The data to be modeled include an asymmetric adjacency matrix $\mathbf{A} = (\mathbf{A}_{ij}) \in \{0,1\}^{n\times n}$, where each binary element $\mathbf{A}_{ij}$ denotes the presence (1) or not (0) of the directed edge from node i to j, and a node attribute matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$. Throughout this paper we assume all edges to be conditionally independent and satisfy $\mathbf{A}_{ij}|\Theta \sim \text{Bernoulli}(p_{ij})$, where Θ is the collection of latent representations and $p_{ij} \in (0,1)$ is the posterior probability to form an edge. The objective is to learn the node representations which can best reconstruct the adjacency matrix.

We propose a deep VAE architecture composed of a GCN encoder and a hierarchical latent space model (HLSM) decoder, as shown in Fig. 1. The encoder block takes the adjacency $\mathbf{A}$ and attribute matrices $\mathbf{X}$, if available, as inputs and learns a hidden state for each node using a directed GCN. Then, the decoder block recursively generates interpretable latent variables from variational posteriors via Monte Carlo sampling and reconstructs the adjacency matrix as outputs. For inference, the hierarchical architecture enables the model to learn variables using not only the priors passed from previous layers of the decoder, but also the approximate likelihood learned by the encoder.

3.1 HLSM DECODER

The HLSM decoder learns interpretable node representations based on the classic LSM approach and generates an adjacency matrix. Assume each node to correspond to a latent position in an unobservable D-dimensional space, denoted as $\boldsymbol{z}_i \in \mathbb{R}^D$. The distances between latent positions indicate the relationships of nodes. The closer two latent positions are, the more possible an edge will exist. Formally, an directed edge is generated by following

$$\mathrm{P}\left(\mathbf{A}_{ij} = 1|\mathbf{X}, \Theta\right) = \sigma\left(\beta_0 - \beta_{out}\|\boldsymbol{\gamma}_i \odot \left(\boldsymbol{z}_i - \boldsymbol{z}_j\right)\| - \beta_{in}\|\boldsymbol{\delta}_j \odot \left(\boldsymbol{z}_i - \boldsymbol{z}_j\right)\|\right), \tag{1}$$

where $\sigma(\cdot)$ is the sigmoid function. The latent positions $\boldsymbol{z}_i$ are involved as Euclidean distances, indicating the relationships between nodes. $\boldsymbol{\gamma}_i$ and $\boldsymbol{\delta}_i \in \mathbb{R}^D$ are the activity and popularity factors,

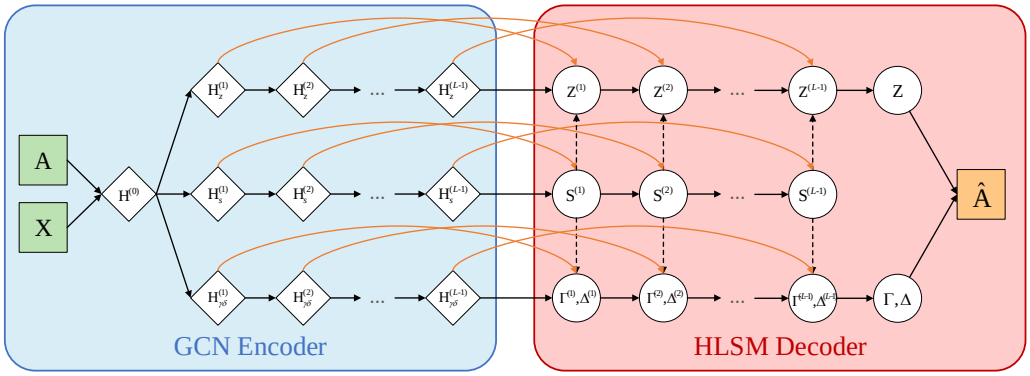


Figure 1: The architecture of hierarchical variational auto-encoder. The HLSM decoder generates a adjacency matrix with latent variables, which are sparsified by the community membership (black dashed arrows). The GCN encoder learns variables using not only the feedforward priors from previous layers (black solid arrows), but also the skipping likelihood of the inputs (orange arcs).

the reverse of which represent the tendencies for a node to send and receive an edge, respectively. The global weights β_{out} and β_{in} are to measure the importance of activity and popularity, respectively, and β_0 is a bias. To be mentioned, though Eq. (1) is designed for directed graphs, it can be easily degraded to an undirected variant by setting $\boldsymbol{\gamma}_i = \boldsymbol{\delta}_i$ and $\beta_{out} = \beta_{in}$.

We build an HLSM decoder to generate three types of latent variables, i.e. the latent positions $\boldsymbol{z}_i^{(l)} \in \mathbb{R}^{G_l}$, the community membership $\boldsymbol{s}_i^{(l)} \in \mathbb{R}^{G_l}$ and the node random factors $\boldsymbol{\gamma}_i$, $\boldsymbol{\delta}_i^{(l)} \in \mathbb{R}^{G_l}$, where G_l is the size of the $l-$th decoder layer. All of these variables are randomly sampled from the variational distributions. At the first layer ($l = 1$), the parameters of variational distributions are defined by priors solely, while for other layers ($l = 2, \ldots, L-1$), the parameters are obtained by the feedforward representations generated from previous layers as well as priors.

Latent position The latent positions $\boldsymbol{z}_i^{(l)}$ represent the location of each node in a latent space, generated as

$$\boldsymbol{z}_i^{(l)} \sim \text{Normal}\left(\boldsymbol{s}_i^{(l)} \odot f(\mathbf{W}_z^{(l-1)}\boldsymbol{z}_i^{(l-1)}), \text{diag}({\boldsymbol{\sigma}_i^{(l)}}^2)\right), \tag{2}$$

where $f(\cdot)$ is a nonlinear activation function (e.g. leaky ReLU), $\odot$ denotes the element-wise multiplication, diag$(\cdot)$ denotes a diagonal matrix, ${\boldsymbol{\sigma}_i^{(l)}}^2 \in \mathbb{R}^{G_l}$ is the prior variance to be specified, and $\mathbf{W}_z^{(l-1)} \in \mathbb{R}^{G_l \times G_{l-1}}$ is a weight matrix to transform the variable from dimension G_{l-1} to G_l.

Community membership The latent positions in Eq. (2) are separated into different communities by the sparse binary variables $\boldsymbol{s}_i^{(l)} = (s_{i1}, \ldots, s_{iG_l})'$, thus an overlapping community structure can be adapted. Referring to Miller et al. (2009), we employ the Indian buffet process (IBP) prior on $\boldsymbol{s}_i^{(l)}$ to learn the effective number of communities given a sufficiently large truncation parameter G_l, i.e.

$$\pi_{ig}^{(l)} = \text{logit}\left(\prod_{h=1}^{g} v_{ih}^{(l)}\right), s_{ig}^{(l)} \sim \text{Bernoulli}\left(\sigma(\pi_{ig}^{(l)})\right). \tag{3}$$

Typically, the log odds $\pi_{ig}^{(l)}$ is generated using a stick-breaking construction, where $v_{ih}^{(l)}$ is drawn from a Beta distribution (Teh et al., 2007). In our model we simplify such hierarchical prior structure by specifying a global v for all nodes. At each layer, the community membership $s_{ig}^{(l)}$ denotes whether node i belongs to community g, hence the size of each decoder layer G_l can be explained as the number of communities. Additionally, the proposed HVAE architecture enables our model to detect community structures at multiple levels of granularities. Letting the layer sizes G_l be downward increasing, the top layers indicate the coarse-grained communities and the bottom layers indicate the fine-grained communities.

Node random factors To model the prevalent power-law of node degrees, we propose the Dirichlet random factors $\boldsymbol{\gamma}_i^{(l)}$, $\boldsymbol{\delta}_i^{(l)} \in \mathbb{R}^{G_l}$, i.e.

$$\boldsymbol{\gamma}_i^{(l)} \sim \text{Dirichlet}\left(\boldsymbol{\xi}_i^{(l)} + \boldsymbol{s}_i^{(l)} \odot f(\mathbf{W}_\gamma^{(l-1)}\boldsymbol{\gamma}_i^{(l-1)})\right), \tag{4}$$

$$\boldsymbol{\delta}_i^{(l)} \sim \text{Dirichlet}\left(\boldsymbol{\psi}_i^{(l)} + \boldsymbol{s}_i^{(l)} \odot f(\mathbf{W}_\delta^{(l-1)}\boldsymbol{\delta}_i^{(l-1)})\right), \tag{5}$$

where $\boldsymbol{\xi}_i^{(l)}$ and $\boldsymbol{\psi}_i^{(l)}$ are prior parameters to be specified, and $\mathbf{W}_\gamma^{(l-1)}$, $\mathbf{W}_\delta^{(l-1)} \in \mathbb{R}^{G_l \times G_{l-1}}$ are weight matrices. Note that the node random factors are dependent on the community membership $\boldsymbol{s}_i^{(l)}$ as well, which can be explained as only the random effects within the community are involved in the representations.

At the output layer ($l = L$), the latent positions $\boldsymbol{z}_i$ and node random factors $\boldsymbol{\gamma}_i$, $\boldsymbol{\delta}_i$ are obtained by linear transformation from dimension G_{L-1} to D. Each column of the weight matrices belongs to a $(D-1)$-simplex. Such transformation also changes the interpretation of the layer size, from the number of communities to the dimension of the latent space. Last, the adjacency matrix is reconstructed with the latent variables by Eq. (1).

3.2 GCN ENCODER

The proposed DLSM employs a deep encoder as a non-iterative recognition model to infer the parameters of posterior distributions. Assuming the mean-field approximation of the variational distributions, the true joint posterior of the latent variables $p_\theta(\Theta|\mathbf{A}, \mathbf{X})$ can be approximated by a variational posterior $q_\phi(\Theta)$, where θ and ϕ denote the generative (decoder) and inference (encoder) parameters to be trained, respectively. Then, the variational posterior is given as

$$\begin{aligned} q_\phi(\boldsymbol{z}, \boldsymbol{s}, \boldsymbol{\gamma}, \boldsymbol{\delta}) = &\prod_{i=1}^{n}\prod_{l=1}^{L} q_\phi(\boldsymbol{z}_i^{(l)}|\boldsymbol{h}_i^{(l)}, \boldsymbol{z}_i^{(l-1)}) q_\phi(\boldsymbol{s}_i|\boldsymbol{h}_i^{(l)}, \boldsymbol{\pi}_i^{(l-1)}) \\ &q_\phi(\boldsymbol{\gamma}_i|\boldsymbol{h}_i^{(l)}, \boldsymbol{\gamma}_i^{(l-1)}) q_\phi(\boldsymbol{\delta}_i|\boldsymbol{h}_i^{(l)}, \boldsymbol{\delta}_i^{(l-1)}), \end{aligned} \tag{6}$$

where $\boldsymbol{h}_i^{(l)} \in \mathbb{R}^{K_l}$ is the output of the encoder and K_l denotes the size of the l-th layer.

GCN has been proved effective in learning the topology of non-Euclidean data, and thus is an ideal choice for the encoder of our model. Referring to Kipf & Welling (2017), we propose the directed GCN operator as

$$\mathbf{H}^{(l+1)} = f\left(\tilde{\mathbf{D}}_{out}\tilde{\mathbf{A}}\tilde{\mathbf{D}}_{in}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right). \tag{7}$$

Here $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_n$ and $\mathbf{I}_n$ is the n-dimensional identity matrix. $\tilde{\mathbf{D}}_{out}$ and $\tilde{\mathbf{D}}_{in}$ are diagonal matrices with elements as the out- and in-degrees of $\tilde{\mathbf{A}}$, respectively. $\mathbf{H}^{(0)} = \mathbf{X}$ if $\mathbf{X}$ is available and $\mathbf{H}^{(0)} = \mathbf{I}_n$ if not. During inference, the hidden states are passed to the corresponding layer of the decoder and then combined with the prior information from previous layers to generate the parameters of variational distributions. Note that the vanilla GCN used here can be substituted by any other GNN for directed graphs.

4 INFERENCE

We now introduce our fast inference method using the SGVB algorithm (Kingma & Welling, 2014a). Compared with the iterative methods such as MCMC adopted by traditional Bayesian random graph approaches, SGVB is much more efficient and scalable. Such method requires differential Monte Carlo expectations to perform backpropagation, thus the reparameterization trick for each of the latent variables is leveraged.

Denoting $\boldsymbol{z}^* \in \mathbb{R}^{G_l}$ a vector with standard Normal elements, the latent positions are reparametrized as $\boldsymbol{z}_i^{(l)} = \hat{\boldsymbol{\mu}}_i^{(l)}(\boldsymbol{z}_i^{(l-1)}, \boldsymbol{s}_i^{(l-1)}, \boldsymbol{h}_i^{(l)}) + \hat{\boldsymbol{\sigma}}_i^{(l)}(\boldsymbol{z}_i^{(l-1)}, \boldsymbol{s}_i^{(l-1)}, \boldsymbol{h}_i^{(l)}) \odot \boldsymbol{z}^*$, where $\hat{\boldsymbol{\mu}}_i^{(l)}(\boldsymbol{z}_i^{(l-1)}, \boldsymbol{s}_i^{(l-1)}, \boldsymbol{h}_i^{(l)})$, $\hat{\boldsymbol{\sigma}}_i^{(l)}(\boldsymbol{z}_i^{(l-1)}, \boldsymbol{s}_i^{(l-1)}, \boldsymbol{h}_i^{(l)})$ are variational posterior parameters.

Following Mehta et al. (2019), the Bernoulli posterior $p_\theta(s_{ig}^{(l)})$ is approximated by the Binary Concrete distribution (Maddison et al., 2017), i.e.

$$\tilde{s}_{ig}^{(l)} = \frac{1}{\lambda}\left(\hat{\pi}_{ig}^{(l)}(\boldsymbol{s}_i^{(l-1)}, \boldsymbol{h}_i^{(l)}) + \text{logit}(u)\right), \tag{8}$$

where $u \sim U(0,1)$, $\hat{\pi}_{ig}^{(l)}(\boldsymbol{s}_i^{(l-1)}, \boldsymbol{h}_i^{(l)})$ is a variational posterior parameter, $\lambda \in \mathbb{R}^+$ is the temperature to be specified, and then $s_{ig}^{(l)} = \sigma(\tilde{s}_{ig}^{(l)})$.

Referring to the recent literature (Joo et al., 2020), we use multiple normalized Gamma variables with unified rate parameters to compose the Dirichlet distributions $p_\theta(\boldsymbol{\gamma}_i^{(l)})$ and $p_\theta(\boldsymbol{\delta}_i^{(l)})$, i.e.

$$\tilde{\boldsymbol{\gamma}}_i^{(l)} \sim \text{Gamma}\left(\hat{\boldsymbol{\xi}}_i^{(l)}(\boldsymbol{\gamma}_i^{(l-1)}, \boldsymbol{s}_i^{(l-1)}, \boldsymbol{h}_i^{(l)}), \mathbf{1}\right), \tag{9}$$

$$\tilde{\boldsymbol{\delta}}_i^{(l)} \sim \text{Gamma}\left(\hat{\boldsymbol{\psi}}_i^{(l)}(\boldsymbol{\delta}_i^{(l-1)}, \boldsymbol{s}_i^{(l-1)}, \boldsymbol{h}_i^{(l)}), \mathbf{1}\right), \tag{10}$$

where $\hat{\boldsymbol{\xi}}_i^{(l)}(\boldsymbol{\gamma}_i^{(l-1)}, \boldsymbol{s}_i^{(l-1)}, \boldsymbol{h}_i^{(l)})$ and $\hat{\boldsymbol{\psi}}_i^{(l)}(\boldsymbol{\delta}_i^{(l-1)}, \boldsymbol{s}_i^{(l-1)}, \boldsymbol{h}_i^{(l)})$ are variational posterior parameters. For the convenience of notations, here the parameters of Gamma distribution are symbolized as vectors, meaning the element-wise operations. The node random factors are then derived by $\boldsymbol{\gamma}_i^{(l)} = \tilde{\boldsymbol{\gamma}}_i^{(l)} / \sum_j^n \tilde{\boldsymbol{\gamma}}_j^{(l)}$, $\boldsymbol{\delta}_i^{(l)} = \tilde{\boldsymbol{\delta}}_i^{(l)} / \sum_j^n \tilde{\boldsymbol{\delta}}_j^{(l)}$. In practice, the Dirichlet variables are magnified by n times to avoid too small values of $\boldsymbol{\gamma}_i^{(l)}$ and $\boldsymbol{\delta}_i^{(l)}$ when n is large.

In particular, the initial variational posterior parameters ($l = 0$) are set as nonlinear combinations of $\boldsymbol{h}_i^{(1)}$ and $\boldsymbol{h}_i^{(L-1)}$, as illustrated in Fig. 1.

The loss function is defined by minimizing the negative evidence lower bound (ELBO), i.e.

$$\begin{aligned}\mathcal{L} = \sum_{i=1}^{n}\sum_{l=1}^{L}\Big(&\text{KL}\left[q_\phi(\boldsymbol{z}_i^{(l)})\middle\|p_\theta(\boldsymbol{z}_i^{(l)})\right] + \text{KL}\left[q_\phi(\boldsymbol{s}_i^{(l)})\middle\|p_\theta(\boldsymbol{s}_i^{(l)})\right] + \text{KL}\left[q_\phi(\boldsymbol{\gamma}_i^{(l)})\middle\|p_\theta(\boldsymbol{\gamma}_i^{(l)})\right] \\ &+\text{KL}\left[q_\phi(\boldsymbol{\delta}_i^{(l)})\middle\|p_\theta(\boldsymbol{\delta}_i^{(l)})\right]\Big) - \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbb{E}_q\left[\log p_\theta(\mathbf{A}_{ij}|\Theta^{(L)})\right],\end{aligned} \tag{11}$$

where $\text{KL}[q(\cdot)||p(\cdot)]$ denotes the Kullback-Leibler (KL) divergence between $q(\cdot)$ and $p(\cdot)$ (of which the full expression is given in Appendix A). The second term is the cross entropy of adjacency matrix reconstruction. Here all of the true posteriors $p_\theta(\Theta^{(l)})$ and variational posteriors $q_\phi(\Theta^{(l)})$, except for the input layer ($l = 1$), are conditioned on $\Theta^{(l-1)}$, which is omitted for simplification.

5 Model Interpretability

In this section, we show that the proposed latent positions and node random factors are interpretable for representing the existence and strength of node influences, by hypothesizing that the true posterior of $\boldsymbol{z}_i$ can be approximated by the variational Normal distribution.

Imagine that there is a new node i with $n_i \geq 2$ neighbors entering a graph and we want to calculate $\boldsymbol{z}_i$ using other existed nodes. It is rational to suppose that the new node should be posited around its neighbors and thus can be denoted as a residual to the mean of its neighbors, i.e.

$$\boldsymbol{z}_i \sim \text{Normal}\left(\bar{\boldsymbol{z}}_i + \boldsymbol{\epsilon}_i, \text{diag}\left(\boldsymbol{\sigma}_i\right)\right), \tag{12}$$

where $\bar{\boldsymbol{z}}_i = \frac{1}{n_i}\sum_{j\in\mathcal{N}_i}\boldsymbol{z}_j$ is the mean position of node i's neighbors, and $\boldsymbol{\epsilon}_i \in \mathbb{R}^D$ is the residual for $\boldsymbol{z}_i$ to the mean position. Intuitively, the residual should depend on all of the neighbors to various extent, of which the influential ones are likely to contribute more. Formally, we decompose the residual as

$$\boldsymbol{\epsilon}_i = \sum_{j\in\mathcal{N}_i} r_{ij}\boldsymbol{e}_{ij}, \tag{13}$$

where $r_{ij} \geq 0$ is a scalar and $\boldsymbol{e}_{ij} \in \mathbb{R}^D$ is a unit vector at the direction of $\boldsymbol{z}_j - \bar{\boldsymbol{z}}_i$. The non-negative scalar denotes the strength of influences for neighbor j exerting on node i (see Fig. 2 in Appendix for

an example in a two-dimensional latent space). An extreme case is $r_{ij} = 0$, which means the latent position of i is not influenced by neighbor j at all.

To further analyze the relation between the latent variables and node influences, we define the prior of r_{ij} as a mixture of a point mass on 0 and an exponential distribution, i.e.

$$f_r(r_{ij}) = \begin{cases} p_0 & \text{if } r_{ij} = 0 \\ (1-p_0) f_{\text{Exp}_\lambda}(r_{ij}) & \text{if } r_{ij} > 0, \end{cases} \tag{14}$$

where $0 < p_0 < 1$ and $f_{\text{Exp}_\lambda}(\cdot)$ denotes the density function of the exponential distribution with parameter λ. What is of our interest is the posterior odds ratio (OR) for $r_{ij} > 0$, defined as $\text{OR}(r_{ij} > 0|\mathbf{A}, \mathbf{X}) = \int_0^\infty f_r(r_{ij} = v|\mathbf{A}, \mathbf{X})/f_r(r_{ij} = 0|\mathbf{A}, \mathbf{X})dv$, which measures the ratio for node influences to exist.

Theorem 1 *Let Θ denote a collection of all parameters except $\boldsymbol{z}_i$. The posterior OR for $r_{ij} > 0$ satisfies*

$$OR(r_{ij} > 0|\mathbf{A}, \mathbf{X}) = \mathbb{E}\left[g\left(\boldsymbol{z}_i, \Theta\right) \middle| r_{ij} = 0, \mathbf{A}, \mathbf{X}\right], \tag{15}$$

where

$$g\left(\boldsymbol{z}_i, \Theta\right) = \frac{(1-p_0)\Phi(u)}{p_0 \phi(u)}, u = \prod_d^D \sigma_{id}^{-1} \left[\left(\boldsymbol{z}_i - \bar{\boldsymbol{z}}_i - \sum_{k \in \mathcal{N}_i, k \neq j} r_{ik} \boldsymbol{e}_{ik} \right)' \boldsymbol{e}_{ij} - \lambda \prod_d^D \sigma_{id}^2 \right], \tag{16}$$

and $\Phi(\cdot)$ and $\phi(\cdot)$ denote the cumulative distribution function and probability density function of the standard Normal distribution, respectively.

The full proof is provided in Appendix B.1. It is interesting that in Theorem 1 $(\boldsymbol{z}_i - \bar{\boldsymbol{z}}_i - \sum_{k \in \mathcal{N}_i, k \neq j} r_{ik} \boldsymbol{e}_{ik})' \boldsymbol{e}_{ij}$ is a scalar projection of $\boldsymbol{z}_i$'s residual component onto the unit vector whose direction is determined by $\boldsymbol{z}_j$. In a word, it shows that the posterior latent positions are able to model the existence of node influences via the residual component projections.

Now we consider the interpretation of node random factors, which explains the strengths of node influences. Formally, we have the following conclusion, i.e.

Theorem 2 *Assume the vectors from $\bar{\boldsymbol{z}}_i$ to all influential neighbors ($r_{ij} > 0$) are orthogonal to each other, and the random factors are the same at all dimensions of the latent space, denoted as γ_i and δ_i, respectively. The strength of influences for neighbor j exerting on node i can be denoted as*

$$r_{ij} = h(\gamma_i, \delta_j, \theta, o) \|\boldsymbol{z}_j - \boldsymbol{z}_i\|, \tag{17}$$

where θ is the angle between $\boldsymbol{z}_i - \bar{\boldsymbol{z}}_i$ and $\boldsymbol{z}_j - \bar{\boldsymbol{z}}_i$, o is the angle between $\bar{\boldsymbol{z}}_i - \boldsymbol{z}_j$ and $\boldsymbol{z}_i - \boldsymbol{z}_j$.

Here $h(\gamma_i, \delta_j, \theta, o)$ is a function of the random factors as well as θ and o (more details are given in Appendix B.2). Theorem 2 reflects that given the direction of $\boldsymbol{e}_{ij}$ and $\boldsymbol{z}_j - \boldsymbol{z}_i$, the node random factors can be regarded as coefficients for measuring the ratio of r_{ij} to $\|\boldsymbol{z}_j - \boldsymbol{z}_i\|$, which means that the strength of influences for a neighbor delivering to the node can be strengthened or undermined via shrinking or enlarging the distance between them by the random factors.

6 EXPERIMENTS

To evaluate the effectiveness and interpretability of the proposed model, we conduct a series of experiments, including link prediction and community detection, on several real-world graphs. The experimental results show that the learned node representations can better fit directed graphs and our model significantly outperforms the baseline methods. The interpretability of node representations are also illustrated through clustering via projecting the latent positions onto a 2D space.

6.1 DATASETS

The experiments of link prediction are conducted on five real-world datasets, namely Emails, Political blogs, Cora, WikiVote and Google. Specifically, the Emails network consists of the members from 42

Table 1: Results (in %) for link prediction on directed graphs. The best results are in **bold** and the second are underlined.

	Emails		Political blogs		Cora		WikiVote		Google	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
LSM	91.3 ± 0.1	90.6 ± 0.1	88.7 ± 0.3	86.5 ± 0.2	86.2 ± 0.2	85.7 ± 0.2	N/A	N/A	N/A	N/A
VGAE	86.9 ± 0.5	85.9 ± 0.6	84.8 ± 0.5	83.0 ± 0.4	72.9 ± 0.3	77.3 ± 0.4	78.6 ± 0.2	77.8 ± 0.2	78.3 ± 0.4	80.0 ± 0.4
SEAL	$\underline{94.2 \pm 0.3}$	$\underline{94.0 \pm 0.5}$	93.0 ± 0.3	$\underline{93.0 \pm 0.4}$	83.2 ± 0.6	86.8 ± 0.5	$\mathbf{97.1 \pm 0.2}$	$\underline{96.8 \pm 0.2}$	$\mathbf{98.8 \pm 0.4}$	$\underline{98.7 \pm 0.4}$
DGLFRM	92.4 ± 0.3	93.1 ± 0.3	88.6 ± 0.4	88.4 ± 0.4	73.7 ± 0.4	78.3 ± 0.6	88.4 ± 0.3	89.8 ± 0.2	86.0 ± 0.0	88.2 ± 0.1
GGVAE	91.5 ± 0.3	90.3 ± 0.6	$\underline{93.6 \pm 0.4}$	92.4 ± 0.5	$\mathbf{91.9 \pm 0.8}$	$\mathbf{92.5 \pm 0.6}$	96.2 ± 0.1	94.7 ± 0.3	97.8 ± 0.3	98.2 ± 0.1
LGVG	93.2 ± 0.2	92.4 ± 0.3	91.4 ± 0.4	90.2 ± 0.3	85.3 ± 0.2	88.2 ± 0.4	94.0 ± 0.0	94.5 ± 0.0	95.7 ± 0.0	95.9 ± 0.0
DGGAN	91.4 ± 0.7	91.0 ± 0.8	91.2 ± 0.6	90.8 ± 0.8	86.1 ± 0.7	88.6 ± 0.6	93.2 ± 0.5	93.4 ± 0.5	92.7 ± 0.4	92.8 ± 0.5
DLSM	$\mathbf{95.2 \pm 0.2}$	$\mathbf{94.1 \pm 0.6}$	$\mathbf{94.8 \pm 0.4}$	$\mathbf{93.7 \pm 0.8}$	$\underline{91.6 \pm 0.2}$	$\mathbf{92.5 \pm 0.2}$	$\underline{97.0 \pm 0.2}$	$\mathbf{97.0 \pm 0.2}$	$\mathbf{98.8 \pm 0.2}$	$\mathbf{98.8 \pm 0.1}$

departments of a European research institution (Leskovec et al., 2007). An edge exists if a person sends at least one email to another. Political blogs is a well-studied social network composed of U.S. political blog nodes (Adamic & Glance, 2005), which are labeled as "liberal" and "conservative" clusters. Two blog pages are connected if one is referenced by another. Cora is a citation network of scientific publications which are classified into seven classes (Sen et al., 2008). Each node and directed edge represent a paper and a citation, respectively. WikiVote is a network of Wikipedia users (Leskovec et al., 2010), where each edge represents a user voting on another to become the administrator. Google is a network of web pages which are connected by hyperlinks (Palla et al., 2007). All of the graphs have been preprocessed by omitting the isolated nodes and loops. In our experiments, the edges are randomly splitted as 85% for training, 10% for testing, and 5% for validation. The descriptive statistics of the datasets are given in Appendix C.

6.2 Baselines

We compare our proposed DLSM[1] with four recent deep generative methods for graph representation learning, i.e. the variational graph auto-encoder (VGAE) (Kipf & Welling, 2016), the deep generative latent feature relational model (DGLFRM) (Mehta et al., 2019), the gravity graph variational auto-encoder (GGVAE) (Salha et al., 2019), the ladder Gamma variational auto-encoder for graphs (LGVG) (Sarkar et al., 2020), and the directed graph generative adversarial network (DGGAN) (Zhu et al., 2021). For link prediction, we also consider a non-generative GNN-based model, i.e. SEAL (Zhang et al., 2021), which is the state-of-the-art method for this task. The methods that are not particularly designed for directed graphs, including VGAE, DGLFRM and LGVG, are slightly modified by altering the original encoder with our proposed directed GCN given in Eq (7).

6.3 Experimental Results

Link Prediction For link prediction, we employ the widely used area under the ROC curve (AUC) and average precision (AP) as evaluation metrics. Hyperparameter settings are provided in Appendix D. The experimental results of DLSM and the baselines are presented in Table 1, where the reported results are the means and standard deviations of 10 independent random splits. On all datasets, our model achieves better or comparable performances to the non-generative method SEAL. However, since it requires to materialize a subgraph for each link, SEAL is much more unscalable than our DLSM and takes about 20 multiples longer for running one epoch. The traditional LSM model is unpractical to fit large graphs such as WikiVote and Google because of the MCMC-based inference methods, while the SGVB method adopted by our model is much more computationally efficient. In addition, it seems that GGVAE performs well on approximately unidirectional graphs (i.e. the reciprocal rate is close to 0 as shown in Table 4) such as Cora, whereas on other more generalized directed graphs where more bidirectional edges exist, our method is significantly superior to GGVAE. This is because our proposed DLSM considers the link reciprocity of directed graphs, which is neglected by GGVAE due to its absolutely asymmetric decoding scheme. We also conduct experiments of link prediction on three real-world undirected graphs, given in Appendix E, which show that our proposed model achieves comparable performances with the baselines and verify the capability for modeling undirected graphs as special cases.

[1]The source code is available at `https://github.com/upperr/DLSM`.

Table 2: Results (in %) for community detection on directed graphs. The best results are in **bold** and the second are underlined.

	Emails		Political blogs		Cora	
	macro F1	micro F1	macro F1	micro F1	macro F1	micro F1
VGAE	31.5 ± 1.6	42.5 ± 1.5	34.9 ± 1.2	52.3 ± 1.4	23.2 ± 1.0	24.8 ± 0.4
DGLFRM	53.3 ± 1.3	60.7 ± 1.0	42.4 ± 1.1	50.5 ± 1.3	18.2 ± 0.5	22.2 ± 0.5
GGVAE	62.8 ± 1.5	67.7 ± 1.4	92.0 ± 1.1	92.5 ± 0.8	22.7 ± 0.7	24.0 ± 0.4
LGVG	67.0 ± 1.3	63.8 ± 1.2	72.8 ± 1.3	73.9 ± 1.3	20.0 ± 0.5	23.9 ± 0.7
DGGAN	62.1 ± 1.6	66.7 ± 1.3	88.9 ± 1.1	89.5 ± 1.3	18.7 ± 1.0	21.4 ± 1.2
DLSM	$\mathbf{72.0 \pm 1.0}$	$\mathbf{77.0 \pm 1.2}$	$\mathbf{92.5 \pm 0.7}$	$\mathbf{92.6 \pm 0.7}$	$\mathbf{24.6 \pm 0.8}$	$\mathbf{25.7 \pm 0.5}$

Table 3: Ablation study results (in %) for link prediction.

	Emails		Political blogs		Cora		WikiVote		Google	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
-S-Δ-Γ	94.3 ± 0.4	93.2 ± 0.4	90.6 ± 0.3	90.2 ± 0.4	86.5 ± 0.5	88.9 ± 0.6	93.1 ± 0.4	92.9 ± 0.3	95.1 ± 0.3	95.3 ± 0.4
-Δ-Γ	93.0 ± 0.5	91.5 ± 0.6	90.1 ± 0.4	87.5 ± 0.5	82.3 ± 0.6	86.0 ± 0.7	87.2 ± 0.6	85.1 ± 0.6	94.9 ± 0.4	95.0 ± 0.3
-Γ	94.9 ± 0.4	93.9 ± 0.5	91.0 ± 0.6	88.1 ± 0.5	85.2 ± 0.8	87.9 ± 0.7	93.6 ± 0.6	93.6 ± 0.5	98.6 ± 0.1	98.5 ± 0.1
-S	94.8 ± 0.5	93.7 ± 0.6	93.6 ± 0.5	91.9 ± 0.6	88.2 ± 0.6	90.9 ± 0.8	96.5 ± 0.3	96.4 ± 0.3	98.4 ± 0.2	98.2 ± 0.1
-HLSM	95.2 ± 0.5	94.0 ± 0.6	93.9 ± 0.4	92.5 ± 0.5	86.8 ± 0.7	86.9 ± 0.6	95.6 ± 0.3	95.8 ± 0.4	98.0 ± 0.3	97.3 ± 0.2
DLSM	95.2 ± 0.2	94.1 ± 0.6	94.8 ± 0.4	93.7 ± 0.8	91.6 ± 0.2	92.5 ± 0.2	97.0 ± 0.2	97.0 ± 0.2	98.8 ± 0.2	98.8 ± 0.1

Community Detection We further conduct community detection on three real-world datasets with ground-truth community labels, i.e. Emails, political blogs and Cora. In particular, for the Emails dataset, communities (departments) with less than 30 nodes are excluded and finally 10 communities are retained. We utilize the node embeddings learned by each model (the latent positions $\boldsymbol{z}_i$ for DLSM) to conduct K-means clustering with ground-truth community numbers. The commonly used macro and micro F1 scores are leveraged as evaluation metrics, which are computed using the most likely mappings between true and predicted clusters. Table 2 shows that our proposed model significantly outperforms all other baselines for graph representation learning. Such results verify the advantages of leveraging the Euclidean distance rather than inner product to compute the divergences between nodes for community detection. (See Appendix G for visualizations of node embeddings using a 2D t-SNE projectionthe distributions of the reverse node random factors, which shows a consistent power-law with node degrees and offers more interpretability.)

Ablation study To justify the effectiveness of each component in our framework, including the community membership $\boldsymbol{s}_i$, node random factors, $\boldsymbol{\delta}_i$, $\boldsymbol{\gamma}_i$, and the HLSM decoder architecture, we compare the proposed DLSM with several variants of our method, which are named by the corresponding ablated components. For example, -Δ-Γ indicates the variant removed node random factors (see Appendix F for more details). The link prediction results are presented in Table 3 and the community detection results are in Table 8 of Appendix. Our DLSM performs best on all datasets for both tasks, verifying that all components are useful under the proposed framework. Interestingly, from the first two lines of Table 3, we find that the community membership seems ineffective when the node random factors are not involved. This reflects that the binary $\boldsymbol{s}_i$ mainly benefits our model by shrinking the redundant latent variables to 0 when there are too many parameters to optimize.

7 Conclusion

We establish a deep generative model to learn multiple highly interpretable node representations for directed graphs. Our proposed DLSM, comprised of a deep GCN encoder and a HLSM decoder, combines the traditional random graphs models with deep learning based methods to take the complementary advantages of interpretability and scalability. Series of experiments have shown that the model is effective for fitting directed graphs and achieves the state-of-the-art performance on link prediction and community detection. In addition, the interpretable node representations learned by the model can naturally represent both the community structure and degree heterogeneity of complex directed graphs. In the future, we shall extend our model for the more complicated scenes such as weighted or dynamic graphs. While the former with multi-valued edges can be simply achieved, the latter demands for a more efficient method to learn the evolutionary topology of graphs.

REFERENCES

Lada A Adamic and Natalie Glance. The political blogosphere and the 2004 US election: divided they blog. In *International Workshop on Link Discovery*, pp. 36–43, 2005.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286 (5439):509–512, 1999.

Giorgio Fagiolo. Clustering in complex directed networks. *Physical Review E*, 76(2):026107, 2007.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864, 2016.

Aditya Grover, Aaron Zweig, and Stefano Ermon. Graphite: Iterative generative modeling of graphs. In *International Conference on Machine Learning*, pp. 2434–2444, 2019.

William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *International Conference on Neural Information Processing Systems*, pp. 1025–1035, 2017.

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

Paul W Holland, Kathryn B Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

Weonyoung Joo, Wonsung Lee, Sungrae Park, and Il-Chul Moon. Dirichlet variational autoencoder. *Pattern Recognition*, 107:107514, 2020.

Brian Karrer and Mark EJ Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83(1):016107, 2011.

Diederik Kingma and Max Welling. Efficient gradient-based inference through transformations between Bayes nets and neural nets. In *International Conference on Machine Learning*, pp. 1782–1790, 2014a.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014b.

Thomas N Kipf and Max Welling. Variational graph auto-encoders. In *NIPS Workshop on Bayesian Deep Learning*, 2016.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Pavel N Krivitsky, Mark S Handcock, Adrian E Raftery, and Peter D Hoff. Representing degree distributions, clustering, and homophily in social networks with latent cluster random effects models. *Social Networks*, 31(3):204–213, 2009.

Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1):2–es, 2007.

Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting positive and negative links in online social networks. In *International Conference on World Wide Web*, pp. 641–650, 2010.

Chris J Maddison, Andriy Mnih, and Yee W Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Nikhil Mehta, Lawrence C Duke, and Piyush Rai. Stochastic blockmodels meet graph neural networks. In *International Conference on Machine Learning*, pp. 4466–4474, 2019.

Kurt Miller, Michael Jordan, and Thomas Griffiths. Nonparametric latent feature models for link prediction. *Advances in Neural Information Processing Systems*, 22:1276–1284, 2009.

Gergely Palla, Illés J Farkas, Péter Pollner, Imre Derényi, and Tamás Vicsek. Directed network modules. *New Journal of Physics*, 9(6):186, 2007.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 701–710, 2014.

Guillaume Salha, Stratis Limnios, Romain Hennequin, Viet-Anh Tran, and Michalis Vazirgiannis. Gravity-inspired graph autoencoders for directed link prediction. In *ACM International Conference on Information and Knowledge Management*, pp. 589–598, 2019.

Arindam Sarkar, Nikhil Mehta, and Piyush Rai. Graph representation learning via ladder Gamma variational autoencoders. In *AAAI Conference on Artificial Intelligence*, pp. 5604–5611, 2020.

Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–93, 2008.

Daniel K Sewell and Yuguo Chen. Latent space models for dynamic networks. *Journal of the American Statistical Association*, 110(512):1646–1657, 2015.

Casper K Sønderby, Tapani Raiko, Lars Maaløe, Søren K Sønderby, and Ole Winther. Ladder variational autoencoders. In *International Conference on Neural Information Processing Systems*, pp. 3745–3753, 2016.

Yee W Teh, Dilan Grür, and Zoubin Ghahramani. Stick-breaking construction for the Indian buffet process. In *Artificial Intelligence and Statistics*, pp. 556–563, 2007.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11), 2008.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. GraphGAN: Graph representation learning with generative adversarial nets. In *AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graph-SAINT: Graph sampling based inductive learning method. In *International Conference on Learning Representations*, 2020.

Muhan Zhang, Pan Li, Yinglong Xia, Kai Wang, and Long Jin. Labeling trick: A theory of using graph neural networks for multi-node representation learning. In *International Conference on Neural Information Processing Systems*, 2021.

Mingyuan Zhou. Infinite edge partition models for overlapping community detection and link prediction. In *International Conference on Artificial Intelligence and Statistics*, pp. 1135–1143, 2015.

Shijie Zhu, Jianxin Li, Hao Peng, Senzhang Wang, and Lifang He. Adversarial directed graph embedding. In *AAAI Conference on Artificial Intelligence*, pp. 4741–4748, 2021.

A Loss Functions

The KL divergence between Normal distributions Kingma & Welling (2014b) for $\boldsymbol{z}_i^{(l)}$ is

$$\mathrm{KL}\left[q_\phi\left(\boldsymbol{z}_i^{(l)}\right)\middle\|p_\theta\left(\boldsymbol{z}_i^{(l)}\right)\right]=\sum_{g=1}^{G_l}-\frac{1}{2}\left(1+\log\left(\hat{\sigma}_{ig}^{(l)}\right)^2-\left(\hat{\mu}_{ig}^{(l)}\right)^2-\left(\hat{\sigma}_{ig}^{(l)}\right)^2\right). \tag{18}$$

The KL divergence between Binary Concrete distributions Maddison et al. (2017) for $\boldsymbol{s}_i^{(l)}$ is

$$\mathrm{KL}\left[q_\phi\left(\boldsymbol{s}_i^{(l)}\right)\middle\|p_\theta\left(\boldsymbol{s}_i^{(l)}\right)\right]=\sum_{g=1}^{G_l}\log\frac{\hat{\pi}_{ig}^{(l)}\left(1+\pi_{ig}^{(l)}e^{-\lambda\tilde{s}_{ig}^{(l)}}\right)^2}{\pi_{ig}^{(l)}\left(1+\hat{\pi}_{ig}^{(l)}e^{-\lambda\tilde{s}_{ig}^{(l)}}\right)^2}, \tag{19}$$

where $\tilde{s}_{ig}^{(l)}$ is the log odds of $s_{ig}^{(l)}$.

The KL divergence between Gamma distributions Joo et al. (2020) for $\boldsymbol{\gamma}_i^{(l)}$ (the same for $\boldsymbol{\delta}_i^{(l)}$) is

$$\mathrm{KL}\left[q_\phi\left(\boldsymbol{\gamma}_i^{(l)}\right)\middle\|p_\theta\left(\boldsymbol{\gamma}_i^{(l)}\right)\right]=\sum_{g=1}^{G_l}\log\frac{\Gamma\left(\xi_{ig}\right)}{\Gamma\left(\hat{\xi}_{ig}\right)}+\left(\hat{\xi}_{ig}-\xi_{ig}\right)\psi\left(\hat{\xi}_{ig}\right), \tag{20}$$

where $\Gamma(\cdot)$ and $\psi(\cdot)$ denote the Gamma and digamma function, respectively.

B Details of Theorems

B.1 Proof of Theorem 1

For any positive value $v>0$, the posterior probability density of r_{ij} at v is

$$\begin{aligned}
&f\left(r_{ij}=v\middle|\mathbf{A},\mathbf{X}\right)\\
=&\iint f\left(r_{ij}=v,\boldsymbol{z}_i,\Theta\middle|\mathbf{A},\mathbf{X}\right)d\boldsymbol{z}_i d\Theta\\
=&\iint\frac{f\left(\mathbf{A},\mathbf{X}\middle|\boldsymbol{z}_i,\Theta\right)f\left(r_{ij}=v,\boldsymbol{z}_i,\Theta\right)}{f\left(\mathbf{A},\mathbf{X}\right)}d\boldsymbol{z}_i d\Theta\\
=&\iint\frac{f\left(\mathbf{A},\mathbf{X}\middle|\boldsymbol{z}_i,\Theta\right)f\left(r_{ij}=0,\boldsymbol{z}_i,\Theta\right)}{f\left(\mathbf{A},\mathbf{X}\right)}\cdot\frac{f\left(r_{ij}=v,\boldsymbol{z}_i,\Theta\right)}{f\left(r_{ij}=0,\boldsymbol{z}_i,\Theta\right)}d\boldsymbol{z}_i d\Theta\\
=&\iint\frac{f\left(r_{ij}=v,\boldsymbol{z}_i,\Theta\right)}{f\left(r_{ij}=0,\boldsymbol{z}_i,\Theta\right)}\cdot f\left(r_{ij}=0,\boldsymbol{z}_i,\Theta\middle|\mathbf{A},\mathbf{X}\right)d\boldsymbol{z}_i d\Theta\\
=&\iint\frac{f\left(\boldsymbol{z}_i\middle|r_{ij}=v,\Theta\right)f\left(r_{ij}=v\right)f\left(\Theta\right)}{f\left(\boldsymbol{z}_i\middle|r_{ij}=0,\Theta\right)f\left(r_{ij}=0\right)f\left(\Theta\right)}\\
&\cdot f\left(r_{ij}=0\middle|\mathbf{A},\mathbf{X}\right)f\left(\boldsymbol{z}_i,\Theta\middle|r_{ij}=0,\mathbf{A},\mathbf{X}\right)d\boldsymbol{z}_i d\Theta.
\end{aligned}\tag{21}$$

According to Eq. (12) and (13), we have

$$\begin{aligned}
&\frac{f\left(\boldsymbol{z}_i\middle|r_{ij}=v,\Theta\right)f\left(r_{ij}=v\right)}{f\left(\boldsymbol{z}_i\middle|r_{ij}=0,\Theta\right)f\left(r_{ij}=0\right)}\\
=&\frac{1-p_0}{p_0}\exp\left\{-\frac{1}{2}\prod_d^D\frac{1}{\sigma_{id}^2}\left[\left(\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i-v\boldsymbol{e}_{ij}-\sum_{k\in\mathcal{N}_i,k\neq j}r_{ik}\boldsymbol{e}_{ik}\right)'\left(\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i-v\boldsymbol{e}_{ij}\right.\right.\right.\\
&\left.\left.\left.-\sum_{k\in\mathcal{N}_i,k\neq j}r_{ik}\boldsymbol{e}_{ik}\right)-\left(\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i-\sum_{k\in\mathcal{N}_i,k\neq j}r_{ik}\boldsymbol{e}_{ik}\right)'\left(\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i-\sum_{k\in\mathcal{N}_i,k\neq j}r_{ik}\boldsymbol{e}_{ik}\right)\right]-\lambda v\right\}\\
=&\frac{1-p_0}{p_0}\exp\left\{-\frac{1}{2}\prod_d^D\frac{1}{\sigma_{id}^2}\left[v^2-2v\left(\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i-\sum_{k\in\mathcal{N}_i,k\neq j}r_{ik}\boldsymbol{e}_{ik}\right)'\boldsymbol{e}_{ij}\right]-\lambda v\right\}\\
=&\frac{1-p_0}{p_0}\exp\left\{-\frac{1}{2}\prod_d^D\frac{1}{\sigma_{id}^2}\left[v-\left(\left(\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i-\sum_{k\in\mathcal{N}_i,k\neq j}r_{ik}\boldsymbol{e}_{ik}\right)'\boldsymbol{e}_{ij}-\lambda\prod_d^D\sigma_{id}^2\right)\right]^2\right.\\
&\left.+\frac{1}{2}\prod_d^D\frac{1}{\sigma_{id}^2}\left[\left(\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i-\sum_{k\in\mathcal{N}_i,k\neq j}r_{ik}\boldsymbol{e}_{ik}\right)'\boldsymbol{e}_{ij}-\lambda\prod_d^D\sigma_{id}^2\right]^2\right\}.
\end{aligned}$$

By dividing $f(r_{ij}=0|\mathbf{A},\mathbf{X})$ on both sides of Eq. (21) and integrating over v on $(0,+\infty)$, we can obtain

$$\begin{aligned}
&\text{OR}(r_{ij}>0|\mathbf{A},\mathbf{X})\\
=&\iint \frac{f\left(\boldsymbol{z}_i\middle|r_{ij}=v,\Theta\right)f\left(r_{ij}=v\right)}{f\left(\boldsymbol{z}_i\middle|r_{ij}=0,\Theta\right)f\left(r_{ij}=0\right)}\cdot f\left(\boldsymbol{z}_i,\Theta\middle|r_{ij}=0,\mathbf{A},\mathbf{X}\right)d\boldsymbol{z}_i d\Theta\\
=&\iint g\left(\boldsymbol{z}_i,\Theta\right)f\left(\boldsymbol{z}_i,\Theta\middle|r_{ij}=0,\mathbf{A},\mathbf{X}\right)d\boldsymbol{z}_i d\Theta\\
=&\mathbb{E}\left[g\left(\boldsymbol{z}_i,\Theta\right)\middle|r_{ij}=0,\mathbf{A},\mathbf{X}\right],
\end{aligned}$$

where

$$g\left(\boldsymbol{z}_i,\Theta\right)=\frac{(1-p_0)\Phi\left(\prod_d^D\sigma_{id}^{-1}\left[\left(\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i-\sum_{k\in\mathcal{N}_i,k\neq j}r_{ik}\boldsymbol{e}_{ik}\right)'\boldsymbol{e}_{ij}-\lambda\prod_d^D\sigma_{id}^2\right]\right)}{p_0\phi\left(\prod_d^D\sigma_{id}^{-1}\left[\left(\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i-\sum_{k\in\mathcal{N}_i,k\neq j}r_{ik}\boldsymbol{e}_{ik}\right)'\boldsymbol{e}_{ij}-\lambda\prod_d^D\sigma_{id}^2\right]\right)}.$$

The conclusion is proved.

B.2 Proof of Theorem 2

We start our proof by giving two definitions, the standard distance (SD) and actual distance (AD) between node i and neighbor j. Consider a simple case, where the influence discrepancies of all neighbors are neglected and the latent position $\boldsymbol{z}_i$ is only dependent on the distances between the node and neighbors (see Fig. 2(b) for an illustration). In this case, the residual $\boldsymbol{\epsilon}_i=\mathbf{0}$ and $\boldsymbol{z}_i$ will coincide with $\bar{\boldsymbol{z}}_i$, thus the distance from neighbor j to node i is equal to $\|\boldsymbol{z}_j-\bar{\boldsymbol{z}}_i\|$, which we refer as the standard distance.

Now we reconsider the influence discrepancies of neighbors upon the node. According to Eq. (1), the actual distance between node i and neighbor j when involving node influences is node influences in our DLSM are involved by rescaling the standard distance. Assume the vectors from the $\bar{\boldsymbol{z}}_i$ to all influential neighbors ($r_{ij}>0$) are orthogonal to each other, and the random factors are the same at all dimensions of the latent space, denoted as γ_i and δ_i, respectively. Then, the actual distance between $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ is

$$\text{AD}_{ij}=(\beta_{out}\gamma_i+\beta_{in}\delta_j)\|\left(\boldsymbol{z}_i-\boldsymbol{z}_j\right)\|. \tag{22}$$

The coefficients $\beta_{out}\gamma_i+\beta_{in}\delta_j$ enlarge or shrink $\text{SD}_{ij}=\|\boldsymbol{z}_j-\bar{\boldsymbol{z}}_i\|$ to AD_{ij} using the node activity and popularity factors. Let $\theta\in(0,\pi/2)$ be the angle between $\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i$ and $\boldsymbol{z}_j-\bar{\boldsymbol{z}}_i$, i.e.

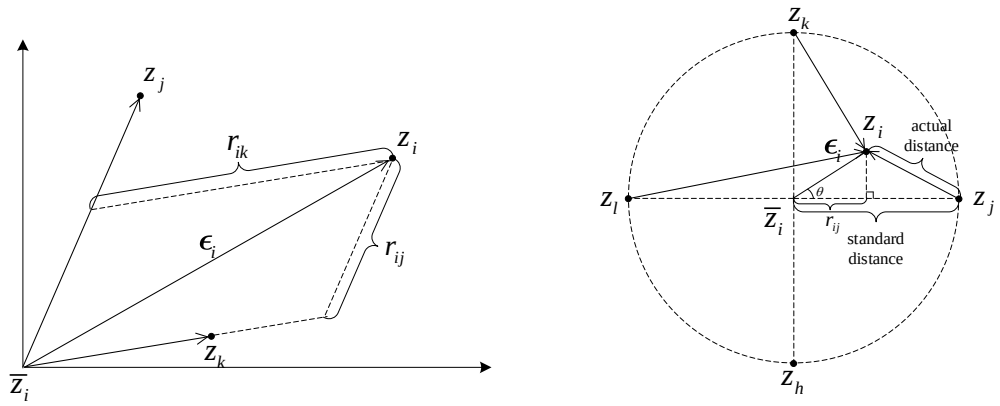
$$\theta=\arccos\left(\frac{(\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i)'(\bar{\boldsymbol{z}}_j-\bar{\boldsymbol{z}}_i)}{\|\boldsymbol{z}_i-\bar{\boldsymbol{z}}_i\|\|\boldsymbol{z}_j-\bar{\boldsymbol{z}}_i\|}\right). \tag{23}$$

Since $\boldsymbol{e}_{ij}$ is orthogonal to other residual components, the modulus of $\boldsymbol{\epsilon}_i$ can be denoted as $\|\boldsymbol{\epsilon}_i\|=r_{ij}/\cos\theta$. According to the law of cosines, we have

$$\begin{aligned}
\text{AD}_{ij}^2=&\text{SD}_{ij}^2+\|\boldsymbol{\epsilon}_i\|^2-2\text{SD}\|\boldsymbol{\epsilon}_i\|\cos\theta\\
=&\text{SD}_{ij}^2+\frac{r_{ij}^2}{\cos^2\theta}-2\text{SD}_{ij}r_{ij}.
\end{aligned} \tag{24}$$

By combining Eq. (22) and Eq. (24), we obtain

$$\begin{aligned}
\|\boldsymbol{z}_j-\bar{\boldsymbol{z}}_i\|^2+\frac{r_{ij}^2}{\cos^2\theta}-2\|\boldsymbol{z}_j-\bar{\boldsymbol{z}}_i\|r_{ij}=&(\beta_{out}\gamma_i+\beta_{in}\delta_j)^2\|\left(\boldsymbol{z}_i-\boldsymbol{z}_j\right)\|^2\\
\left(\frac{r_{ij}}{\cos\theta}-\cos\theta\|\boldsymbol{z}_j-\bar{\boldsymbol{z}}_i\|\right)^2=&\left[(\beta_{out}\gamma_i+\beta_{in}\delta_j)^2-\sin^2\theta\right]\|\left(\boldsymbol{z}_i-\boldsymbol{z}_j\right)\|^2\\
\frac{r_{ij}}{\cos\theta}-\cos\theta\|\boldsymbol{z}_j-\bar{\boldsymbol{z}}_i\|=&\pm\left[(\beta_{out}\gamma_i+\beta_{in}\delta_j)^2-\sin^2\theta\right]^{\frac{1}{2}}\|\boldsymbol{z}_j-\boldsymbol{z}_i\|.
\end{aligned}$$



(a) Decomposion of the residual

(b) Standard and actual distances

Figure 2: (a) An illustration for the residual decomposition with two influential neighbors (the influence strengths for other neighbors are assumed to be 0). The residual for $\boldsymbol{z}_i$ can be decomposed as components at the direction of neighbor $\boldsymbol{z}_j$ and $\boldsymbol{z}_k$, and r_{ij}, r_{ik} are the corresponding influence strengths. (b) An example for the standard and actual distances in a two-dimensional latent space. The vectors from the mean position $\bar{\boldsymbol{z}}_i$ to influential neighbors, i.e. $\boldsymbol{z}_j$ and $\boldsymbol{z}_k$, are orthogonal and equal length.

Given $\theta \in (0, \pi/2)$ (when $\theta \in (\pi/2, \pi)$, $r_{ij} = 0$), the sign for the right side depends on the positive or negative correlation for $\|\boldsymbol{\epsilon}_i\|$ and AD given θ, which is determined by the angle between $\bar{\boldsymbol{z}}_i - \boldsymbol{z}_j$ and $\boldsymbol{z}_i - \boldsymbol{z}_j$, denoted as

$$\begin{aligned} o &= \arccos\left(\frac{(\beta_{out}\gamma_i + \beta_{in}\delta_j)\,(\boldsymbol{z}_i - \boldsymbol{z}_j)'(\bar{\boldsymbol{z}}_i - \boldsymbol{z}_j)}{(\beta_{out}\gamma_i + \beta_{in}\delta_j)\,\|\boldsymbol{z}_i - \boldsymbol{z}_j\|^2}\right) \\ &= \arccos\left(\frac{\bar{\boldsymbol{z}}_i - \boldsymbol{z}_j}{\boldsymbol{z}_i - \boldsymbol{z}_j}\right). \end{aligned} \tag{25}$$

The sign is positive when $o \in (0, \pi/2 - \theta)$ and negative when $o \in (\pi/2 - \theta, \pi)$.

In summary, we have the following conclusion, i.e.

$$r_{ij} = h(\gamma_i, \delta_j, \theta, o)\|\boldsymbol{z}_j - \boldsymbol{z}_i\|,$$

where

$$h(\gamma_i, \delta_j, \theta, o) = \begin{cases} \cos^2\theta - \cos\theta\left[(\beta_{out}\gamma_i + \beta_{in}\delta_j)^2 - \sin^2\theta\right]^{\frac{1}{2}}, & \text{if } o \in \left(0, \frac{\pi}{2} - \theta\right) \\ \cos^2\theta + \cos\theta\left[(\beta_{out}\gamma_i + \beta_{in}\delta_j)^2 - \sin^2\theta\right]^{\frac{1}{2}}, & \text{if } o \in \left(\frac{\pi}{2} - \theta, \pi\right), \end{cases}$$

and θ, o is defined by Eq. (23) and Eq. (25), respectively.

C Datasets for Directed Link Prediction

The descriptive statistics of the directed graph datasets for link prediction and community detection are summarized in Table 4.

D Hyperparameter Setting and Sensitivity Analysis

The proposed DLSM employs a hierarchical VAE architecture composed of a GCN encoder and HLSM decoder. In practice, both of the two parts have three layers, and the layer sizes are fixed to be 32/64/128 and 50/100/50 for the encoder and decoder, respectively. Note that the last layer of the

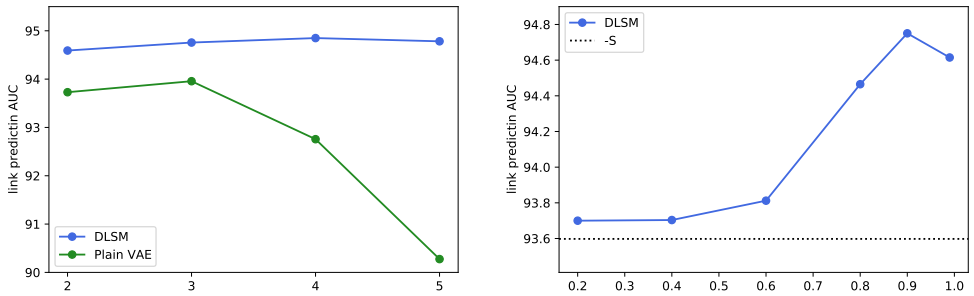
Table 4: Descriptive statistics of the real graph datasets. $|\mathcal{V}|$ and $|\mathcal{E}|$ are the numbers of nodes and edges, respectively, CC is the clustering coefficient (Fagiolo, 2007), d_{max}^{out} and d_{max}^{in} are the maximal in-degree and out-degree of all nodes, respectively, d_{avg} is the average degree of all nodes (average in-degree equals to average out-degree), ED $= \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{A}_{ij}/(n(n-1))$ is the edge density, and RR $= \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{A}_{ij}a_{ji}/\sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{A}_{ij}$ is the reciprocal rate.

Dataset	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	CC	d_{max}^{out}	d_{max}^{in}	d_{avg}	ED	RR
Emails	986	24,929	0.4124	333	211	25.3	0.0257	0.7112
Political blogs	1,222	19,021	0.2459	256	337	15.6	0.0127	0.2426
Cora	2,708	5,429	0.1600	5	166	2.0	0.0007	0.0556
WikiVote	7,115	103,689	0.0896	893	457	14.6	0.0020	0.0565
Google	15,763	171,206	0.4007	852	11,397	10.8	0.0007	0.2541

decoder ($l = L$) is a linear full connection layer which transforms the sparse latent variables to dense for adjacency reconstruction. Fig. 3(a) shows the link prediction results for increasing numbers of layers, and the performance of our method is continuously rising until more than 4 layers.

The HLSM stochastic decoder requires prior distributions to generate latent variables. Specifically, we employ a standard Normal prior for the latent positions $\boldsymbol{z}_i^{(l)}$, i.e. the variance $\boldsymbol{\sigma}_i^{(l)} = 1$. For the community membership $\boldsymbol{s}_i^{(l)}$, we use the IBP prior to infer community numbers given the truncation G_l (i.e. layer sizes of the decoder) and a stick-breaking parameters $v \in (0, 1)$. Typically, a smaller v results in fewer active communities (i.e. non-zero elements of $\boldsymbol{s}_i^{(l)}$). Fig. 3(b) presents the link prediction results of our method with different values of v on the Political blogs dataset, and our method performs best when $v \in (0.9, 1)$. Therefore, we set $v = 0.9$ to ensure the model fully exploring the potential community structure of graphs. Last, we adopt flat priors for the Dirichlet node random factors $\boldsymbol{\gamma}_i$ and $\boldsymbol{\delta}_i$, where all parameters are set as $\xi_{ig}^{(l)} = \psi_{ig}^{(l)} = 1/G_l$.

For inference, the temperature parameter λ of the Binary Concrete distribution is fixed to 1. All models are trained by 1,000-2,000 iterations with a learning rate of 0.01 on an RTX 2080 Ti GPU.



(a) Number of layers

(b) Stick-breaking parameter v

Figure 3: Sensitivity analysis for link prediction on the Political blogs dataset. (a) The plain VAE (green) tends to collapse as the number of layers increases, while the performance of DLSM (blue) is much more stable and continuously rises until more than 4 layers. (b) The performance of DLSM (blue solid line) reaches peak when v is about 0.9, and is consistently better than that of the ablated variant -S (black dashed line).

E Link Prediction on Undirected Graphs

Undirected graphs can be regarded as special cases in our framework, as an undirectional edge is treated as a bidirectional one. In this section we provide some experimental results for link prediction

Table 5: Descriptive statistics of the real-world undirected graph datasets. $|\mathcal{V}|$ and $|\mathcal{E}|$ are the numbers of nodes and edges, respectively, CC is the clustering coefficient, d_{max} is the maximal node degrees, d_{avg} is the average degree of all nodes, ED is the edge density, defined as ED $= 2|\mathcal{E}|/(|\mathcal{V}|(|\mathcal{V}|-1))$, and FD is the dimension of attribute features (0 denotes no features available).

Dataset	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	CC	d_{max}	d_{avg}	ED	FD
NIPS12	2,037	3,134	0.7463	45	3.1	0.0015	0
Cora	2,708	5,278	0.3893	168	3.9	0.0014	1,433
Pubmed	19,717	88,648	0.1117	171	4.5	0.0002	500

Table 6: Results (in %) for link prediction on undirected graphs. The best results are in **bold** and the second are underlined.

	NIPS12		Cora		Pubmed	
	AUC	AP	AUC	AP	AUC	AP
LSM	85.7 ± 1.5	86.4 ± 0.6	86.1 ± 0.7	87.2 ± 0.5	N/A	N/A
VGAE	87.9 ± 0.6	91.1 ± 0.4	92.6 ± 0.0	93.3 ± 0.0	94.2 ± 0.8	93.9 ± 0.9
SEAL	$\mathbf{90.7 \pm 0.4}$	92.5 ± 0.5	$\mathbf{94.5 \pm 0.6}$	$\mathbf{95.2 \pm 0.6}$	$\mathbf{97.0 \pm 0.5}$	$\underline{96.5 \pm 0.6}$
DGLFRM	87.3 ± 0.3	90.1 ± 0.3	93.4 ± 0.2	93.8 ± 0.2	94.0 ± 0.0	95.0 ± 0.4
GGVAE	89.5 ± 0.8	90.7 ± 0.7	89.2 ± 0.4	90.4 ± 0.6	93.4 ± 0.5	94.1 ± 0.4
LGVG	90.1 ± 0.3	$\underline{92.6 \pm 0.7}$	$\underline{93.6 \pm 0.4}$	$\underline{95.0 \pm 0.6}$	94.6 ± 0.3	95.6 ± 0.2
DGGAN	89.9 ± 0.5	91.3 ± 0.6	93.3 ± 0.6	94.2 ± 0.6	94.0 ± 0.2	94.6 ± 0.3
DLSM	$\underline{90.5 \pm 0.1}$	$\mathbf{92.7 \pm 0.1}$	93.1 ± 0.5	93.5 ± 0.5	$\underline{96.9 \pm 0.2}$	$\mathbf{96.7 \pm 0.1}$

on three real-world undirected graph datasets. The baselines are the same as those in the experiments of directed graphs.

E.1 Datasets

We consider three real-world undirected graph datasets, namely NIPS12, Cora and Pubmed. NIPS12 is a coauthor network of the authors in NIPS papers from volumes 1-12 Zhou (2015). Cora is the same dataset used in the experiments for directed link prediction, except that the directions of edges are neglected here. Pubmed is a citation network of scientific publications Sen et al. (2008). Both Cora and Pubmed datasets contain sparse bag-of-words feature matrices, which are used as node attributes. The descriptive statistics of the datasets are summarized in Table 5.

E.2 Link prediction results

The results of link prediction on undirected graphs are presented in Table 6. On all datasets, our DLSM achieves the best or comparative performances with the state-of-the-art baselines for undirected graphs.

F More Details for Ablation Study

We consider five variants of our proposed model. Here we provide more details about these variants. -HLSM indicates to replace the HLSM decoder of our model using a plain VAE with the same numbers of layers. Others are simplified variants by removing the corresponding components, as presented in Table 7. In particular, -Δ refers to set $\boldsymbol{\gamma}_i = \boldsymbol{\delta}_i$ (thus the model has degenerated to an undirected form yet still involves degree heterogeneity of graphs), and -S-Δ-Γ can be viewed as a variant of VGAE by changing the plain decoder with our proposed HLSM architecture.

The community detection results of ablation study is given in Table 8. Generally, models including the community membership perform better than those without this component, and the full version DLSM reaches the best performances on all datasets.

Table 7: Components of ablation variants. A circle denotes the component is involved.

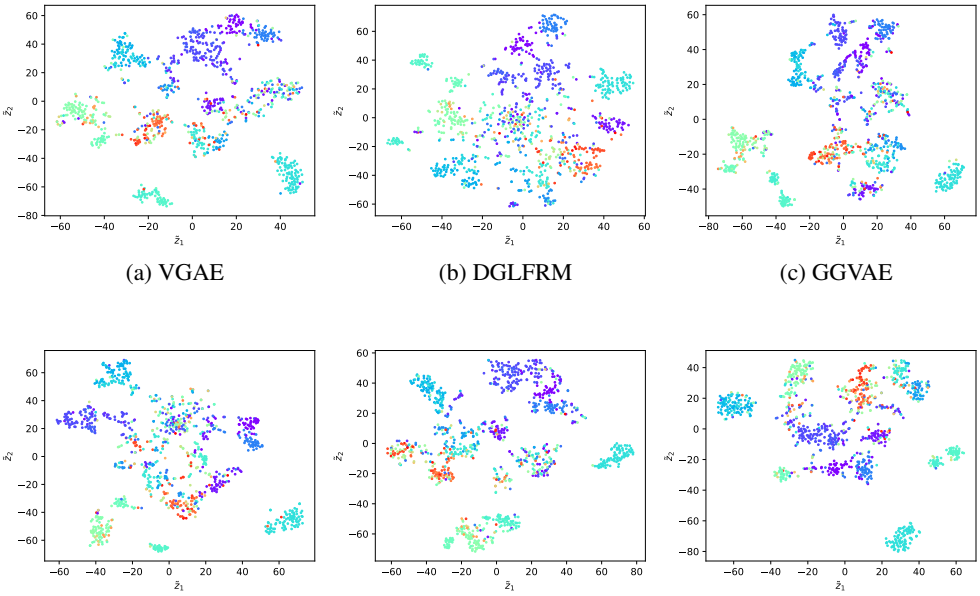
	HLSM	$\boldsymbol{z}_i$	$\boldsymbol{s}_i$	$\boldsymbol{\delta}_i$	$\boldsymbol{\gamma}_i$
-S-Δ-Γ	○	○			
-Δ-Γ	○	○	○		
-Γ	○	○	○	○	
-S	○	○		○	○
-HLSM		○	○	○	○
DLSM	○	○	○	○	○

Table 8: Ablation study results (in %) for community detection.

	Emails		Political blogs		Cora	
	macro F1	micro F1	macro F1	micro F1	macro F1	micro F1
-S-Δ-Γ	64.8 ± 1.7	67.8 ± 1.5	89.6 ± 0.6	90.7 ± 0.5	19.0 ± 0.3	18.9 ± 0.4
-Δ-Γ	66.5 ± 1.2	70.4 ± 1.2	90.9 ± 0.6	91.0 ± 0.6	22.5 ± 0.5	23.4 ± 0.4
-Γ	69.6 ± 1.3	74.1 ± 1.2	92.4 ± 0.4	92.3 ± 0.5	22.7 ± 0.6	24.0 ± 0.5
-S	65.3 ± 1.4	68.5 ± 1.5	91.1 ± 0.5	91.3 ± 0.7	20.9 ± 0.3	21.6 ± 0.4
-HLSM	70.0 ± 1.1	72.4 ± 1.2	90.6 ± 0.6	90.1 ± 0.6	23.5 ± 0.4	24.6 ± 0.5
DLSM	72.0 ± 1.0	77.0 ± 1.2	92.5 ± 0.7	92.6 ± 0.7	24.6 ± 0.8	25.7 ± 0.5

G VISUALIZATIONS OF NODE REPRESENTATIONS

Latent position We leverage a 2D t-SNE projection (Van der Maaten & Hinton, 2008) to visualize the learned latent positions for the Emails dataset. As comparisons, we also illustrate the node representations learned by DGLFRM and LGVG, both of which also consider the community structure but overlook the degree heterogeneity of graphs. The transformed latent variables learned by these three models are plotted in Fig. 4. It is clearly seen that our model performs best in fitting such directed graphs.

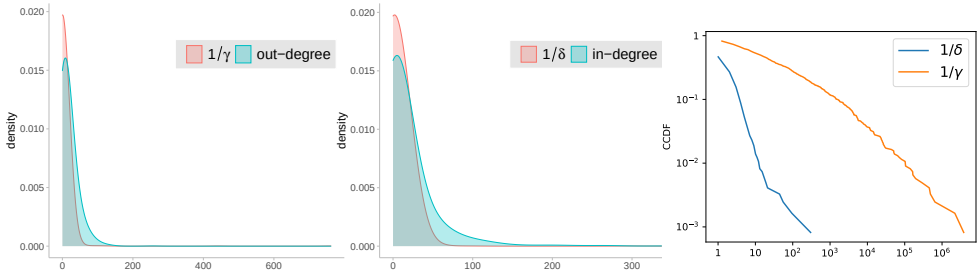


(a) VGAE (b) DGLFRM (c) GGVAE

(d) LGVG (e) DGGAN (f) DLSM

Figure 4: Visualizations of the latent positions learned on the Email network using a 2D t-SNE projection. Points in different colors denote nodes from the ground-truth communities.

Reverse node random factors The pairwise node random factors $\boldsymbol{\gamma}_i$ and $\boldsymbol{\delta}_i$ are supposed to measure the heterogeneity of out-degrees and in-degrees, respectively, which typically follow the power-law (Barabási & Albert, 1999). Fig. 5(a) and (b) present the probability density distributions (PDD) of the node degrees and reverse random factors learned by DLSM on the political blogs network. It seems that the degree distributions are well fitted by the reverse node random factors, indicating that our DLSM can well represent the degree heterogeneity via these latent variables. Furthermore, Fig. 5(c) illustrates the complementary cumulative distributions (CCD) of the random factors. As can be seen, the logarithm CCD of both $\boldsymbol{\gamma}_i$ and $\boldsymbol{\delta}_i$ are approximately linear, with different slopes though. This shows that the proposed Dirichlet latent variables are flexible enough to accommodate the power-law distribution of degrees.



(a) PDD of out-degrees (b) PDD of in-degrees (c) CCD of random factors

Figure 5: Probability density distributions of the degrees and node random factors learned by DLSM.