struc2gauss: Structural role preserving network embedding via Gaussian embedding

Citation for published version:
Pei, Y, Du, X, Zhang, J, Fletcher, G & Pechenizkiy, M 2020, 'struc2gauss: Structural role preserving network embedding via Gaussian embedding', *Data Mining and Knowledge Discovery*, vol. 34, no. 4, pp. 1072-1103. https://doi.org/10.1007/s10618-020-00684-x

Digital Object Identifier (DOI):
10.1007/s10618-020-00684-x

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Publisher's PDF, also known as Version of record

Published In:
Data Mining and Knowledge Discovery

Data Mining and Knowledge Discovery (2020) 34:1072–1103
https://doi.org/10.1007/s10618-020-00684-x

struc2gauss: Structural role preserving network embedding via Gaussian embedding

Yulong Pei[1] · Xin Du[1] · Jianpeng Zhang[2] · George Fletcher[1] · Mykola Pechenizkiy[1]

Received: 30 May 2019 / Accepted: 10 April 2020 / Published online: 12 May 2020

Abstract

Network embedding (NE) is playing a principal role in network mining, due to its ability to map nodes into efficient low-dimensional embedding vectors. However, two major limitations exist in state-of-the-art NE methods: **role preservation** and **uncertainty modeling**. Almost all previous methods represent a node into a point in space and focus on local structural information, i.e., neighborhood information. However, neighborhood information does not capture global structural information and point vector representation fails in modeling the uncertainty of node representations. In this paper, we propose a new NE framework, *struc2gauss*, which learns node representations in the space of Gaussian distributions and performs network embedding based on global structural information. *struc2gauss* first employs a given node similarity metric to measure the global structural information, then generates structural context for nodes and finally learns node representations via Gaussian embedding. Different structural similarity measures of networks and energy functions of Gaussian embedding are investigated. Experiments conducted on real-world networks demonstrate that *struc2gauss* effectively captures global structural information while state-of-the-art network embedding methods fail to, outperforms other methods on the structure-based clustering and classification task and provides more information on uncertainties of node representations.

Keywords Gaussian embedding · Structural similarity · Uncertainty modeling

Responsible editor: Evangelos Papalexakis.

✉ Yulong Pei
y.pei.1@tue.nl

Extended author information available on the last page of the article

1 Introduction

Network analysis consists of numerous tasks including community detection (Fortunato 2010), role discovery (Rossi and Ahmed 2015), link prediction (Liben-Nowell and Kleinberg 2007), etc. As relations exist between nodes that disobey the *i.i.d* assumption, it is non-trivial to apply traditional data mining techniques in networks directly. Network embedding (NE) fills the gap by mapping nodes in a network into a low-dimensional space according to their structural information in the network. It has been reported that using embedded node representations can achieve promising performance on many network analysis tasks (Cao et al. 2015; Grover and Leskovec 2016; Perozzi et al. 2014; Ribeiro et al. 2017).

Previous NE techniques mainly relied on eigendecomposition (Shaw and Jebara 2009; Tenenbaum et al. 2000), but the high computational complexity of eigendecomposition makes it difficult to apply in real-world networks. With the fast development of neural network techniques, unsupervised embedding algorithms have been widely used in natural language processing (NLP) where words or phrases from the vocabulary are mapped to vectors in the learned embedding space, e.g., *word2vec* (Mikolov et al. 2013a, b) and GloVe (Pennington et al. 2014). By drawing an analogy between paths consists of several nodes on networks and word sequences in text, DeepWalk (Perozzi et al. 2014) learns node representations based on random walks using the same mechanism of *word2vec*. Afterwards, a sequence of studies have been conducted to improve DeepWalk either by extending the definition of neighborhood to higher-order proximity (Cao et al. 2015; Grover and Leskovec 2016; Perozzi et al. 2016; Tang et al. 2015b) or incorporating more information for node representations such as attributes (Li et al. 2017; Wang et al. 2017) and heterogeneity (Chang et al. 2015; Tang et al. 2015a).

Although a variety of NE methods have been proposed, two major limitations exist in previous NE studies: **role preservation** and **uncertainty modeling**. Previous methods focused only on one of these two limitations and while neglecting the other. In particular, for **role preservation**, most studies applied random walk to learn representations. However, random walk based embedding strategies and their higher-order extensions can only capture local structural information, i.e., first-order and higher-order proximity within the neighborhood of the target node (Lyu et al. 2017). Local structural information is reflected in community structures of networks. But these methods may fail in capturing global structural information, i.e., structural roles (Rossi and Ahmed 2015; Pei et al. 2018). Global structural information represents roles of nodes in networks, where two nodes have the same role if they are structurally similar from a global perspective. An example of global structural information (roles) and local structural information (communities) is shown in Fig. 1. In summary, nodes that belong to the same community require dense local connections while nodes that have the same role may have no common neighbors at all (Tu et al. 2018). Empirical evidence based on this example for illustrating this limitation will be shown in Sect. 5.2. For **uncertainty modeling**, most previous methods represented a node into a point vector in the learned embedding space. However, real-world networks may be noisy and imbalanced. For example, node degree distributions in real-world networks are often skewed where some low-degree nodes may contain less discriminative information (Tu et al. 2018). Point vector representations learned by these methods are

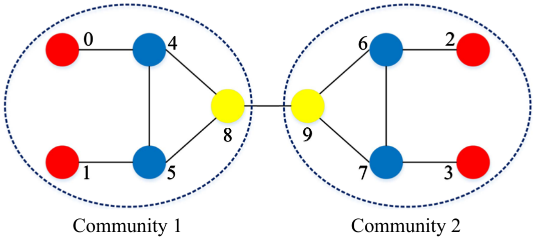


Fig. 1 An example of ten nodes belonging to (1) three groups (different colors indicate different groups) based on global structural information, i.e., the structural roles and (2) two groups (groups are shown by the dashed ellipses) based on local structural information, i.e., the communities. For example, nodes 0, 1, 4, 5 and 8 belong to the same group Community 1 based on local structural perspective because they have more internal connections. Node 0 and 2 are far from each other, but they are in the same group based on global structural perspective (Color figure online)

deterministic (Dos Santos et al. 2016) and are not capable of modeling the uncertainties of node representations.

There are a few studies trying to address these limitations in the literature. For instance, *struc2vec* (Ribeiro et al. 2017) builds a hierarchy to measure similarity at different scales, and constructs a multilayer graph to encode the structural similarities. *SNS* (Lyu et al. 2017) discovers graphlets as a pre-processing step to obtain the structural similar nodes. DRNE (Tu et al. 2018) learns network embedding by modeling regular equivalence (Wasserman and Faust 1994). However, these studies aim only to solve the problem of **role preservation** to some extent. Thus the limitation of **uncertainty modeling** remains a challenge. Dos Santos et al. (2016) and Bojchevski and Günnemann (2017) put effort in improving classification tasks by embedding nodes into Gaussian distributions but both methods only capture the neighborhood information based on random walk techniques. DVNE (Zhu et al. 2018) learns Gaussian embedding for nodes in the Wasserstein space as the latent representations to capture uncertainties of nodes, but they focus only on first- and second-order proximity of networks same to previous methods. Therefore, the problem of **role preservation** has not been solved in these studies.

In this paper, we propose *struc2gauss*, a new structural role preserving network embedding framework. *struc2gauss* learns node representations in the space of Gaussian distributions and performs NE based on global structural information so that it can address both limitations simultaneously. On the one hand, *struc2gauss* generates node context based on a global structural similarity measure to learn node representations so that global structural information can be taken into consideration. On the other hand, *struc2gauss* learns node representations via Gaussian embedding and each node is represented as a Gaussian distribution where the mean indicates the position of this node in the embedding space and the covariance represents its uncertainty. Furthermore, we analyze and compare two different energy functions for Gaussian embedding to calculate the closeness of two embedded Gaussian distributions, i.e., expected likelihood and KL divergence. To investigate the influence of structural information, we also compare *struc2gauss* to two other structural similarity measures for networks, i.e., MatchSim and SimRank.

We summarize the contributions of this paper as follows:

- We propose a flexible structure preserving network embedding framework, *struc2gauss*, which learns node representations in the space of Gaussian distributions. *struc2gauss* is capable of preserving structural roles and modeling uncertainties.
- We investigate the influence of different energy functions in Gaussian embedding and compare to different structural similarity measures in preserving global structural information of networks.
- We conduct extensive experiments in node clustering and classification tasks which demonstrate the effectiveness of *struc2gauss* in capturing the global structural role information of networks and modeling the uncertainty of learned node representations.

The rest of the paper is organized as follows. Section 2 provides an overview of the related work. We present the problem statement in Sect. 3. Section 4 explains the technical details of *struc2gauss*. In Sect. 5 we then discuss our experimental study. The possible extensions of *struc2gauss* are discussed in Sect. 6. Finally, in Sect. 7 we draw conclusions and outline directions for future work.

2 Related work

2.1 Network embedding

Network embedding methods map nodes in a network into a low-dimensional space according to their structural information in the network. The learned node representations can boost performance in many network analysis tasks, e.g., community detection and link prediction. Previous methods mainly viewed NE as part of dimensionality reduction techniques (Goyal and Ferrara 2018). They first construct a pairwise similarity graph based on neighborhood and then embed the nodes of the graph into a lower dimensional vector space. Locally Linear Embedding (LLE) (Tenenbaum et al. 2000) and Laplacian Eigenmaps (Belkin and Niyogi 2001) are two representative methods in this category. SPE (Shaw and Jebara 2009) learns a low-rank kernel matrix to capture the structures of input graph via a set of linear inequalities as constraints. But the high computational complexity makes these methods difficult to apply in real-world networks.

With increasing attention attracted by neural network research, unsupervised neural network techniques have opened up a new world for embedding. *word2vec* as well as Skip-Gram and CBOW (Mikolov et al. 2013a, b) learn low-rank representations of words in text based on word context and show promising results of different NLP tasks. Based on *word2vec*, DeepWalk (Perozzi et al. 2014) first introduces such embedding mechanism to networks by treating nodes as words and random walks as sentences. Afterwards, a sequence of studies have been conducted to improve DeepWalk either by extending the definition of neighborhood to higher-order proximity (Cao et al. 2015; Grover and Leskovec 2016; Perozzi et al. 2016; Tang et al. 2015b) or incorporating more information for node representations such as attributes (Li et al. 2017;

Wang et al. 2017) and heterogeneity (Chang et al. 2015; Tang et al. 2015a). Recently, deeper neural networks have also been introduced in NE problem to capture the nonlinear characteristics of networks, such as SDNE (Wang et al. 2016). However, these approaches represent a node into a point vector in the learned embedding space and are not capable of modeling the uncertainties of node representations. To solve this problem, inspired by Vilnis and McCallum (2014), Gaussian embedding has been used in NE. Bojchevski and Günnemann (2017) learns node embeddings by leveraging Gaussian embedding to capture uncertainties. Dos Santos et al. (2016) combines Gaussian embedding and classification loss function for multi-label network classification. DVNE (Zhu et al. 2018) learns a Gaussian embedding for each node in the Wasserstein space as the latent representation so that the uncertainties can be modeled. We refer the reader to Hamilton et al. (2017b), Cui et al. (2018) and Cai et al. (2018) for more details.

Recent years have witnessed increasing interest in neural networks on graphs. Graph neural networks (Scarselli et al. 2008) can also learn node representations but using more complicated operations such as convolution. Kipf and Welling (2016) proposes a GCN model using an efficient layer-wise propagation rule based on a first-order approximation of spectral convolutions on graphs. Gilmer et al. (2017) introduces a general message passing neural network framework to interpret different previous neural models for graphs. GraphSAGE (Hamilton et al. 2017a) learns node representations in an inductive manner sampling a fixed-size neighborhood of each node, and then performing a specific aggregator over it. Embedding Propagation (EP) (Duran and Niepert 2017) learns representations of graphs by passing messages forward and backward in an unsupervised setting. Graph Attention Networks (GATs) (Velickovic et al. 2017) extend graph convolutions by utilizing masked self-attention layers to assign different importances to different nodes with different sized neighborhoods.

However, most NE methods as well as graph neural networks only concern the local structural information represented by paths consists of linked nodes, i.e., the community structures of networks. But they fail to capture global structural information, i.e., structural roles. SNS (Lyu et al. 2017), *struc2vec* (Ribeiro et al. 2017) and DRNE (Tu et al. 2018) are exceptions which take global structural information into consideration. SNS uses graphlet information for structural similarity calculation as a pre-propcessing step. *struc2vec* applies the dynamic time warping to measure similarity between two nodes' degree sequences and builds a new multilayer graph based on the similarity. Then similar mechanism used in DeepWalk has been used to learn node representations. DRNE explicitly models regular equivalence, which is one way to define the structural role, and leverages the layer normalized LSTM (Ba et al. 2016) to learn the representations for nodes. Another related work focusing on global structural information is REGAL (Heimann et al. 2018). REGAL aims at matching nodes across different graphs so the global structural patterns should be considered. However, its target is network alignment but not representation learning. A brief summary of these NE methods is list in Table 1.

Table 1 A brief summary of different NE methods

Method	Community (local)	Role (global)	Uncertainty
DeepWalk (Perozzi et al. 2014)	√		
LINE (Tang et al. 2015b)	√		
GraRep (Cao et al. 2015)	√		
PTE (Tang et al. 2015a)	√		
Walklets (Perozzi et al. 2016)	√		
node2vec (Grover and Leskovec 2016)	√		
EP (Duran and Niepert 2017)	√		
GraphSage (Hamilton et al. 2017a)	√		
struc2vec (Ribeiro et al. 2017)		√	
DRNE (Tu et al. 2018)		√	
GraphWave (Donnat et al. 2018)		√	
DVNE (Zhu et al. 2018)	√		√
SNS (Lyu et al. 2017)		√	
Our method		√	√

Note that (1) we only list NE methods for homogeneous networks without attributes, and (2) *node2vec* (Grover and Leskovec 2016) aims to capture both local and global structure information but walk-based sampling strategy does not effectively capture global structure information, as shown in our experiments in Sect. 5

2.2 Structural similarity

Structure based network analysis tasks can be categorized into two types: structural similarity calculation and network clustering .

Calculating structural similarities between nodes is a hot topic in recent years and different methods have been proposed. SimRank (Jeh and Widom 2002) is one of the most representative notions to calculate structural similarity. It implements a recursive definition of node similarity based on the assumption that two objects are similar if they relate to similar objects. SimRank++ (Antonellis et al. 2008) adds an evidence weight which partially compensates for the neighbor matching cardinality problem. P-Rank (Zhao et al. 2009) extends SimRank by jointly encoding both in- and out-link relationships into structural similarity computation. MatchSim (Lin et al. 2009) uses maximal matching of neighbors to calculate the structural similarity. RoleSim (Jin et al. 2011) is the only similarity measure which can satisfy the automorphic equivalence properties.

Network clusters can be based on either global or local structural information. Graph clustering based on global structural information is the problem of role discovery (Rossi and Ahmed 2015). In social science research, roles are represented as concepts of equivalence (Wasserman and Faust 1994). Graph-based methods and feature-based methods have been proposed for this task. Graph-based methods take nodes and edges as input and directly partition nodes into groups based on their structural patterns. For example, Mixed Membership Stochastic Blockmodel (Airoldi et al. 2008) infers the role distribution of each node using the Bayesian generative model.

Feature-based methods first transfer the original network into feature vectors and then use clustering methods to group nodes. For example, RolX (Henderson et al. 2012) employs ReFeX (Henderson et al. 2011) to extract features of networks and then uses non-negative matrix factorization to cluster nodes. Local structural information based clustering corresponds to the problem of community detection (Fortunato 2010). A community is a group of nodes that interact with each other more frequently than with those outside the group. Thus, it captures only local connections between nodes.

3 Problem statement

We illustrated local community structure and global role structure in Sect. 1 using the example in Fig. 1. In this section, definitions of community and role will be presented and then we formally define the problem of structural role preserving network embedding.

Structural role is from social science and used to describe nodes in a network from a global perspective. Formally,

Definition 1 (*Structural role*) In a network, a set of nodes have the same role if they share similar structural properties (such as degree, clustering coefficient, and betweenness) and structural roles can often be associated with various functions in a network.

For example, hub nodes with high degree in a social network are more likely to be opinion leaders, whereas bridge nodes with high betweenness are gatekeepers to connect different groups. Structural roles can reflect the global structural information because two nodes which have the same role could be far from each other and have no direct links or shared neighbors. In contrast to roles, community structures focus on local connections between nodes.

Definition 2 (*Community structure*) In a network, communities can represent the local structures of nodes, i.e., the organization of nodes in communities, with many edges joining nodes of the same community and comparatively few edges joining nodes of different communities (Fortunato 2010). A community is a set of nodes where nodes in this set are densely connected internally.

It can be seen that the focus of community structure is the internal and local connections so it aims to capture the local structural information of networks

In this study, we only consider the global structural information, i.e., structural role information, so without mentioning it explicitly, structural information indicates the global one and the keyphrases "structural role information" and "global structural information" are used interchangeably.

Definition 3 (*Structural Role Preserving Network Embedding*) Given a network $G = (V, E)$, where V is a set of nodes and E is a set of edges between the nodes, the problem of **Structural Preserving Network Embedding** aims to represent each node $v \in V$ into a Gaussian distribution with mean μ and covariance Σ in a low-dimensional space $\mathbb{R}^d$, i.e., learning a function

$$f : V \to \mathcal{N}(x; \mu, \Sigma),$$

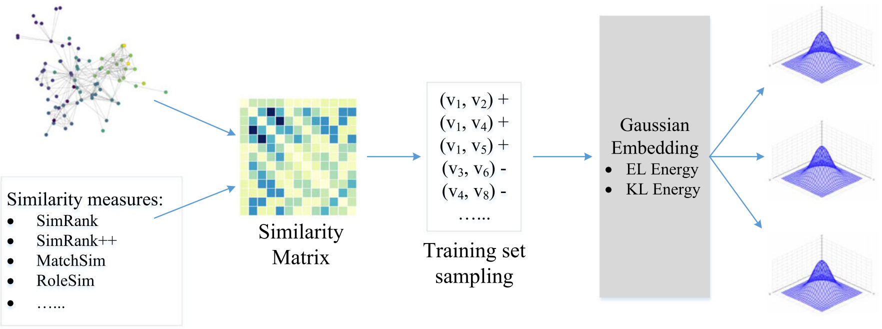


Fig. 2 Overview of the *struc2gauss* framework. *struc2gauss* consists of three components: similarity calculation, training set sampling and Gaussian embedding

where $\mu \in \mathbb{R}^d$ is the mean, $\Sigma \in \mathbb{R}^{d\times d}$ is the covariance and $d \ll |V|$. In the space $\mathbb{R}^d$, the global structural role information of nodes introduced in Definition 1 can be preserved, i.e., if two nodes have the same role their means should be similar, and the uncertainty of node representations can be captured, i.e., the values of variances indicate the levels of uncertainties of learned representations.

4 *struc2gauss*

An overview of our proposed *struc2gauss* framework is shown in Fig. 2. Given a network, a similarity measure is employed to calculate the similarity matrix, then the training set which consists of positive and negative pairs are sampled based on the similarity matrix. Finally, Gaussian embedding techniques are applied on the training set and generate the embedded Gaussian distributions as the node representations and uncertainties of the representations. Besides, we analyze the computational complexity and the flexibility of our *struc2gauss* framework.

4.1 Structural similarity calculation

It has been theoretically proved that random walk sampling based NE methods are not capable of capturing structural equivalence (Lyu et al. 2017) which is one way to model the structural roles in networks (Wasserman and Faust 1994). Thus, to capture the global structural information, we calculate the pairwise structural similarity as a pre-processing step similar to Lyu et al. (2017) and Ribeiro et al. (2017).

In the literature, a variety of structural similarity measures have been proposed to calculate node similarity based on the structures of networks, e.g., SimRank (Jeh and Widom 2002), MatchSim (Lin et al. 2009) and RoleSim (Jin et al. 2011, 2014). However, not all of these measures can capture the global structural role information and we will show the empirical evidence in the experiments in Sect. 5. Therefore, in this paper we leverage RoleSim for the structural similarity since it satisfies all the requirements of Axiomatic Role Similarity Properties for modeling the equivalence (Jin et al. 2011), i.e., the structural roles. RoleSim also generalizes Jaccard coefficient and corresponds linearly to the maximal weighted matching. RoleSim similarity between two nodes u and v is defined as:

$$RoleSim(u, v) = (1 - \beta) \max_{M(u,v)} \frac{\sum_{(x,y)\in M(u,v)} RoleSim(x, y)}{|N(u)| + |N(v)| - |M(u, v)|} + \beta \qquad (1)$$

where $|N(u)|$ and $|N(v)|$ are the numbers of neighbors of node u and v, respectively. $M(u, v)$ is a matching between $N(u)$ and $N(v)$, i.e., $M(u, v) \subseteq N(u) \times N(v)$ is a bijection between $N(u)$ and $N(v)$. The parameter β is a decay factor where $0 < \beta < 1$. The intuition of RoleSim is that two nodes are structurally similar if their corresponding neighbors are also structurally similar. This intuition is consistent with the notion of automorphic and regular equivalence (Wasserman and Faust 1994).

In practice, RoleSim values can be computed iteratively and are guaranteed to converge. The procedure of computing RoleSim consists of three steps:

- Step 1: Initialize matrix of RoleSim scores R^0;
- Step 2: Compute the kth iteration R^k scores for the $(k-1)$th iteration's values, R^{k-1} using:

$$R^k(u, v) = (1 - \beta) \max_{M(u,v)} \frac{\sum_{(x,y)\in M(u,v)} R^{k-1}(x, y)}{|N(u)| + |N(v)| - |M(u, v)|} + \beta \qquad (2)$$

- Step 3: Repeat Step 2 until R values converge for each pair of nodes.

Note that there are other strategies can be used to capture the global structural role information except structural similarity, and these possible strategies will be discussed in Sect. 6. The advantage of RoleSim in capturing structural roles to other structural measures will also be discussed empirically in Sect. 5.6.

4.2 Training set sampling

The target of structural role preserving network embedding is to map nodes in the network to a latent space where the learned latent representations of two nodes are (1) more similar if these two nodes are structurally similar, and (2) more dissimilar if these two nodes are not structurally similar. Hence, we need to generate structurally similar and dissimilar node pairs as the training set based on the similarity we learned in Sect. 4.1. We name the structurally similar pairs of nodes the positive set and the structurally dissimilar pairs the negative set.

In detail, for node v, we rank its similarity values towards other nodes and then select top-k most similar nodes $u_i, i = 1, \ldots, k$ to form its positive set $\Gamma_+ = \{(v, u_i)|i = 1, \ldots, k\}$. For the negative set, we randomly select the same number of nodes $\{u'_i, i = 1, \ldots, k\}$ same to Vilnis and McCallum (2014) and other random walk sampling based methods (Grover and Leskovec 2016; Tang et al. 2015b; Perozzi et al. 2014), i.e., $\Gamma_- = \{(v, u'_i)|i = 1, \ldots, k\}$. Therefore, k is a parameter indicating the *number of positive/negative nodes per node*. We will generate r positive and negative sets for each node where r is a parameter indicating the *number of samples per node*. The influence of these parameters will be analyzed empirically in Sect. 5.7. Note that the selection of the positive set is similar to that in DeepWalk and the difference is that we follow the similarity rank to select the positive nodes instead of random walks.

4.3 Gaussian embedding

4.3.1 Overview

Recently language modeling techniques such as *word2vec* have been extensively used to learn word representations in and almost all NE studies are based on these word embedding techniques. However, these NE studies map each entity to a fixed point vector in a low-dimension space so that the uncertainties of learned embeddings are ignored. Gaussian embedding aims to solve this problem by learning density-based distributed embeddings in the space of Gaussian distributions (Vilnis and McCallum 2014). Gaussian embedding has been utilized in different graph mining tasks including triplet classification on knowledge graphs (He et al. 2015), multi-label classification on heterogeneous graphs (Dos Santos et al. 2016) and link prediction and node classification on attributed graphs (Bojchevski and Günnemann 2017).

Gaussian embedding trains with a ranking-based loss based on the ranks of positive and negative samples. Following Vilnis and McCallum (2014), we choose the max-margin ranking objective which can push scores of positive pairs above negatives by a margin defined as:

$$\mathcal{L} = \sum_{(v,u)\in\Gamma_+} \sum_{(v',u')\in\Gamma_-} \max(0, m - \mathcal{E}(z_v, z_u) + \mathcal{E}(z_{v'}, z_{u'})) \tag{3}$$

where Γ_+ and Γ_- are the positive and negative pairs, respectively. $\mathcal{E}(\cdot,\cdot)$ is the energy function which is used to measure the similarity of two distributions, z_v and z_u are the learned Gaussian distributions for nodes v and u, and m is the margin separating positive and negative pairs. In this paper, we present two different energy functions to measure the similarity of two distributions for node representation learning, i.e., expected likelihood and KL divergence based energy functions. For the learned Gaussian distribution $z_i \sim \mathcal{N}(0; \mu_i, \Sigma_i)$ for node i, to reduce the computational complexity, we restrict the covariance matrix Σ_i to be diagonal and spherical in this work.

4.3.2 Expected likelihood based energy

Although both dot product and inner product can be used to measure similarity between two distributions, dot product only considers means and does not incorporate covariances. Thus, we use inner product to measure the similarity. Formally, the integral of inner product between two Gaussian distributions z_i and z_j (learned Gaussian embeddings for node i and j respectively), a.k.a., expected likelihood, is defined as:

$$E(z_i, z_j) = \int_{x\in\mathbb{R}} \mathcal{N}(x; \mu_i, \Sigma_i)\mathcal{N}(x; \mu_j, \Sigma_j)dx = \mathcal{N}(0; \mu_i - \mu_j, \Sigma_i + \Sigma_j). \tag{4}$$

For simplicity in computation and comparison, we use the logarithm of Eq. (4) as the final energy function:

$$\begin{aligned}\mathcal{E}_{EL}(z_i, z_j) &= \log E(z_i, z_j) = \log \mathcal{N}(0; \mu_i - \mu_j, \Sigma_i + \Sigma_j)\\ &= \frac{1}{2}\Big\{(\mu_i - \mu_j)^T(\Sigma_i + \Sigma_j)^{-1}(\mu_i - \mu_j) + \log\det(\Sigma_i + \Sigma_j) + d\log(2\pi)\Big\}\end{aligned} \tag{5}$$

where d is the number of dimensions. The gradient of this energy function with respect to the means μ and covariances Σ can be calculated in a closed form as:

$$\begin{aligned}\frac{\partial\mathcal{E}_{EL}(z_i, z_j)}{\partial\mu_i} &= -\frac{\partial\mathcal{E}(z_i.z_j)_{EL}}{\partial\mu_j} = -\Delta_{ij}\\ \frac{\partial\mathcal{E}_{EL}(z_i, z_j)}{\partial\Sigma_i} &= \frac{\partial\mathcal{E}(z_i.z_j)_{EL}}{\partial\Sigma_j} = \frac{1}{2}(\Delta_{ij}\Delta_{ij}^T - (\Sigma_i + \Sigma_j)^{-1}),\end{aligned} \tag{6}$$

where $\Delta_{ij} = (\Sigma_i + \Sigma_j)^{-1}(\mu_i - \mu_j)$ (He et al. 2015; Vilnis and McCallum 2014). Note that expected likelihood is a symmetric similarity measure, i.e., $\mathcal{E}_{EL}(z_i, z_j) = \mathcal{E}_{EL}(z_j, z_i)$.

4.3.3 KL divergence based energy

KL divergence is another straightforward way to measure the similarity between two distributions so we utilize the energy function $\mathcal{E}_{KL}(z_i, z_j)$ based on the KL divergence to measure the similarity between Gaussian distributions z_i and z_j (learned Gaussian embeddings for node i and j respectively):

$$\begin{aligned}\mathcal{E}_{KL}(z_i, z_j) &= D_{KL}(z_i, z_j)\\ &= \int_{x\in\mathbb{R}} \mathcal{N}(x; \mu_i, \Sigma_i)\log\frac{\mathcal{N}(x; \mu_j, \Sigma_j)}{\mathcal{N}(x; \mu_i, \Sigma_i)}dx\\ &= \frac{1}{2}\Big\{tr(\Sigma_i^{-1}\Sigma_j) + (\mu_i - \mu_j)^T\Sigma_i^{-1}(\mu_i - \mu_j) - \log\frac{\det(\Sigma_j)}{\det(\Sigma_i)} - d\Big\}\end{aligned} \tag{7}$$

where d is the number of dimensions. Similarly, we can compute the gradients of this energy function with respect to the means μ and covariances Σ:

$$\begin{aligned}\frac{\partial\mathcal{E}_{KL}(z_i, z_j)}{\partial\mu_i} &= -\frac{\partial\mathcal{E}_{KL}(z_i.z_j)}{\partial\mu_j} = -\Delta'_{ij}\\ \frac{\partial\mathcal{E}_{KL}(z_i, z_j)}{\partial\Sigma_i} &= \frac{1}{2}\left(\Sigma_i^{-1}\Sigma_j\Sigma_i^{-1} + \Delta'_{ij}\Delta_{ij}'^T - \Sigma_i^{-1}\right)\\ \frac{\partial\mathcal{E}_{KL}(z_i, z_j)}{\partial\Sigma_j} &= \frac{1}{2}\left(\Sigma_j^{-1} - \Sigma_i^{-1}\right)\end{aligned} \tag{8}$$

where $\Delta'_{ij} = \Sigma_i^{-1}(\mu_i - \mu_j)$.

Note that KL divergence based energy is asymmetric but we can easily extend to a symmetric similarity measure as follows:

$$\mathcal{E}(z_i, z_j) = \frac{1}{2}(D_{KL}(z_i, z_j) + D_{KL}(z_j, z_i)). \tag{9}$$

4.4 Learning

To avoid the means to grow too large and ensure the covariances to be positive definite as well as reasonably sized, we regularize the means and covariances to learn the embedding (Vilnis and McCallum 2014). Due to the different geometric characteristics, two different hard constraint strategies have been used for means and covariances, respectively. Note that we only consider diagonal and spherical covariances. In particular, we have

$$\|\mu_i\| \leq C, \ \forall i \tag{10}$$

$$c_{min} I \prec \Sigma_i \prec c_{max} I, \ \forall i. \tag{11}$$

The constraint on means guarantees them to be sufficiently small and constraint on covariances ensures that they are positive definite and of appropriate size. For example, $\Sigma_{ii} \leftarrow \max(c_{min}, \min(c_{max}, \Sigma_{ii}))$ can be used to regularize diagonal covariances.

We use AdaGrad (Duchi et al. 2011) to optimize the parameters. The learning procedure is described in Algorithm 1. Initialization phase is from line 1 to 4, context generation is shown in line 7, and Gaussian embeddings are learned from line 8 to 14.

Algorithm 1 The Learning Algorithm of *struc2gauss*

```
Input: An energy function $\mathcal{E}(z_i, z_j)$, a graph $G = (V, E)$, embedding dimension $d$, constraint values $C$
    for mean and $c_{max}$ and $c_{min}$ for covariance, learning rate $\alpha$, and maximum epochs $n$.
Output: Gaussian embeddings (mean vector $\mu$ and covariance matrix $\Sigma$) for nodes $v \in V$
1: for all $v \in V$ do
2:     Initialize mean $\mu$ for $v$
3:     Initialize covariance $\Sigma$ for $v$
4:     Regularize $\mu$ and $\Sigma$ with constraint in Eq. (10) and (11)
5: end for
6: while not reach the maximum epochs $n$ do
7:     Generate positive and negative sets $\Gamma_+$ and $\Gamma_-$ for each node
8:     if use expected likelihood based energy then
9:         Update means and covariances based on Eq. (6)
10:    end if
11:    if use KL divergence based energy then
12:        Update means and covariances based on Eq. (8)
13:    end if
14:    Regularize $\mu$ and $\Sigma$ with constraint in Eq. (10) and (11)
15: end while
```

4.5 Computational complexity

The complexity of different components of *struc2gauss* are analyzed as follows:

1. For structural similarity calculation using RoleSim, the computational complexity is $O(kn^2d)$, where n is the number of nodes, k is the number of iterations and d is the average of $y \log y$ over all node-pair bipartite graph in G (Jin et al. 2011) where $y = |N(u)| \times |N(v)|$ for each pair of nodes u and v. The complexity $O(y \log y)$ is from the complexity of the fast greedy algorithm offers a $\frac{1}{2}$-approximation of the globally optimal matching.
2. To generate the training set based on similarity matrix, we need to sample from the most similar nodes for each node, i.e., to select k largest numbers from an unsorted array. Using heap, the complexity is $O(n \log k)$.
3. For Gaussian embedding, the operations include matrix addition, multiplication and inversion. In practice, as stated above, we only consider two types of covariance matrices, i.e., diagonal and spherical, so all these operations have the complexity of $O(n)$.

Overall, the component of similarity calculation is the bottleneck of the framework. One possible and effective way to optimize this part is to set the similarity to be 0 if two nodes have a large difference in degrees. The reason is: (1) we generate the context only based on most similar nodes; and (2) two nodes are less likely to be structural similar if their degrees are very different.

5 Experiments

We evaluate *struc2gauss* in different tasks in order to understand its effectiveness in capturing structural information, capability in modeling uncertainties of embeddings and stability of the model towards parameters. We also study the influence of different similarity measures empirically. The source code of *struc2gauss* is available online.[1]

5.1 Experimental setup

5.1.1 Datasets

We conduct experiments on two types of network datasets: networks with and without ground-truth labels where these labels can represent the global structural role information of nodes in the networks. For networks with labels, to compare to state-of-the-art, we use air-traffic networks from Ribeiro et al. (2017) where the networks are undirected, nodes are airports, edges indicate the existence of commercial flights and labels correspond to their levels of activities. For networks without labels, we select five real-world networks in different domains from Network Repository.[2] A brief introduction to these datasets is shown in Table 2. Note that the numbers of groups for networks

[1] https://bitbucket.org/paulpei/struc2gauss/src/master/.

[2] http://networkrepository.com/index.php.

Table 2 A brief introduction to data sets

Type	Dataset	# nodes	# edges	# groups
With labels	Brazilian-air	131	1038	4
	European-air	399	5995	4
	USA-air	1190	13,599	4
Without labels	Arxiv GR-QC	5242	28,980	8
	Advogato	6551	51,332	11
	Hamsterster	2426	16,630	10
	Anybeat	12,645	67,053	15
	Epinion	26,588	100,120	18

without labels are determined by Minimum Description Length (MDL) (Henderson et al. 2012).

5.1.2 Baselines

We compare *struc2gauss* with several state-of-the-art NE methods.

- DeepWalk (Perozzi et al. 2014): DeepWalk (Perozzi et al. 2014) learns node representations based on random walks using the same mechanism of *word2vec* by drawing an analogy between paths consists of several nodes on networks and word sequences in text. The structural information is captured by the paths of nodes generated by random walks.
- *node2vec* (Grover and Leskovec 2016): It extends DeepWalk to learn latent representations from the node paths generated by biased random walk. Two hyperparameters p and q are used to control the random walk to be breadth-first or depth-first. In this way, *node2vec* can capture the structural information in networks. Note that when $p = q = 1$, *node2vec* degrades to DeepWalk.
- LINE (Tang et al. 2015b): It learns node embeddings via preserving both the local and global network structures. By extending DeepWalk, LINE aims to capture both the first-order, i.e., the neighbors of nodes, and second-order proximities, i.e., the shared neighborhood structures of nodes.
- Embedding Propagation (EP) (Duran and Niepert 2017): EP is an unsupervised learning framework for network embedding and learns vector representations of graphs by passing two types of messages between neighboring nodes. EP, as one of graph neural networks, is similar to graph convolutional networks (GCN) (Kipf and Welling 2016). The difference is that EP is unsupervised and GCN is designed for semi-supervised learning.
- *struc2vec* (Ribeiro et al. 2017): It learns latent representations for the structural identity of nodes. Due to its high computational complexity, we use the combination of all optimizations proposed in the paper for large networks.
- *graph2gauss* (Vilnis and McCallum 2014): It maps each node into a Gaussian distribution where the mean indicates the position of a node in the embedded space and the covariance denotes the uncertainty of the learned representation.

Bojchevski and Günnemann (2017) and Dos Santos et al. (2016) extend the original Gaussian embedding method to network embedding task.

- DRNE (Tu et al. 2018): It learns node representations based on the concept of regular equivalence. DRNE utilizes a layer normalized LSTM to represent each node by aggregating the representations of their neighborhoods in a recursive way so that the global structural information can be preserved.
- GraphWave (Donnat et al. 2018): It leverages heat wavelet diffusion patterns to learn a multidimensional structural embedding for each node based on the diffusion of a spectral graph wavelet centered at the node. Then the wavelets as distributions are used to capture structural similarity in graphs.

For all baselines, we use the implementation released by the original authors. For our framework *struc2gauss*, we test four variants: *struc2gauss* with expected likelihood and diagonal covariance (*s2g*_el_d), expected likelihood and spherical covariance (*s2g*_el_s), KL divergence and diagonal covariance (*s2g*_kl_d), and KL divergence and spherical covariance (*s2g*_kl_s). Note that we only use means of Gaussian distributions as the node embeddings in role clustering and classification tasks. The covariances are left for uncertainty modeling.

For other settings including parameters and evaluation metrics, different settings will be discussed in each task.

5.2 Case study: visualization in 2-D space

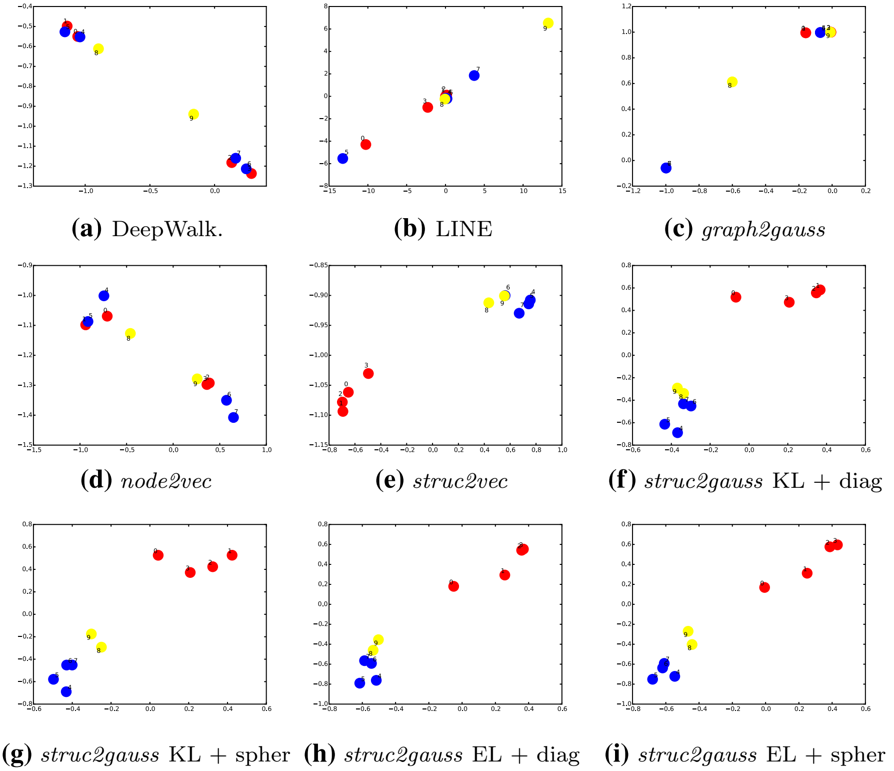
We use the toy example shown in Fig. 1 to demonstrate the effectiveness of *struc2gauss* in capturing the global structural information and the failure of other state-of-the-art techniques in this task. The toy network consists of ten nodes and they can be clustered from two different perspectives:

- from the perspective of the global role structure, they belong to three groups, i.e., {0, 1, 2, 3} (yellow color), {4, 5, 6, 7} (blue color) and {8, 9} (red color) because different groups have different structural functions in this network;
- from the perspective of the local community structure, they belong to two groups, i.e., {0, 1, 4, 5, 6, 8} and {2, 3, 6, 7, 9} because there are denser connections/more edges inside each community that outside the community.

Note that from the perspective of role discovery, these three groups of nodes can be explained to play the roles of *periphery*, *star* and *bridge*, respectively.

In this study, we aim to preserve the global structural information in network embedding. Figure 3 shows the learned node representations by different methods. For shared parameters in all methods, we use the same settings by default: representation dimension: 2, number of walks per node: 20, walk length: 80, skipgram window size: 5. For *node2vec*, we set $p = 1$ and $q = 2$. For *graph2gauss* and *struc2gauss*, the number of walks per node is 20 and the number of positive/negative nodes per node is 5. The constraint for means C is 2 and constraints for covariances c_{min} and c_{max} are 0.5 and 2, respectively. From the visualization results, it can be observed that:

- Our proposed *struc2gauss* outperforms all other methods. Both diagonal and spherical covariances can separate nodes based on global structural information and



(a) DeepWalk. **(b)** LINE **(c)** *graph2gauss*

(d) *node2vec* **(e)** *struc2vec* **(f)** *struc2gauss* KL + diag

(g) *struc2gauss* KL + spher **(h)** *struc2gauss* EL + diag **(i)** *struc2gauss* EL + spher

Fig. 3 Latent representations in $\mathbb{R}^2$ learned by **a** DeepWalk, **b** LINE, **c** GraRep, **d** *node2vec*, **e** *struc2vec*, **f** *struc2gauss* using KL divergence with diagonal covariance, **g** *struc2gauss* using KL divergence with spherical covariance, **g** *struc2gauss* using KL divergence with diagonal covariance, **h** *struc2gauss* using expected likelihood with diagonal covariance, and **i** *struc2gauss* using expected likelihood with spherical covariance

struc2gauss with spherical covariances performs better than diagonal covariances since it can recognize *star* and *bridge* nodes better.

- Methods aim to capture the global structural information performs better than random walk sampling based methods. For example, *struc2vec* can solve this problem to some extent. However, there is overlap between node 6 and 9. It has been stated that *node2vec* can capture the structural equivalence but the visualization shows that it still captures the local structural information similar to DeepWalk.
- DeepWalk, LINE and *graph2gauss* fail to capture the global structural information because these methods are based on random walk which only captures the local community structures. DeepWalk is capable to capture the local structural information since nodes are separated into two parts corresponding to the two communities shown in Fig. 1.

5.3 Structural role clustering

The most common network mining application based on global structural information is the problem of *role discovery* and role discovery essentially is a clustering task. Thus, we consider this task to illustrate the potential of node representations learned by *struc2gauss*. We use the latent representations learned by different methods (in *struc2gauss*, we use means of learned Gaussian distribution) as features and K-means as the clustering algorithm to cluster nodes.

Parameters For these baselines, we use the same settings in the literature: representation dimension: 128, number of walks per node: 20, walk length: 80, skipgram window size: 10. For *node2vec*, we set $p = 1$ and $q = 2$. For *graph2gauss* and *struc2gauss*, we set the constraint for means C to be 2 and constraints for covariances c_{min} and c_{max} to be 0.5 and 2, respectively. The number of walks per node is 10, the number of positive/negative nodes per node is 120 and the representation dimension is also 128.

Evaluation metrics To quantitatively evaluate clustering performance in labeled networks, we use *Normalized Mutual Information (NMI)* as the evaluation metric. NMI is obtained by dividing the mutual information by the arithmetic average of the entropy of obtained cluster and ground-truth cluster. It evaluates the clustering quality based on information theory, and is defined by normalization on the mutual information between the cluster assignments and the pre-existing input labeling of the classes:

$$NMI(\mathcal{C}, \mathcal{D}) = \frac{2 * \mathcal{I}(\mathcal{C}, \mathcal{D})}{\mathcal{H}(\mathcal{C}) + \mathcal{H}(\mathcal{D})}, \tag{12}$$

where obtained cluster $\mathcal{C}$ and ground-truth cluster $\mathcal{D}$. The mutual information $\mathcal{I}(\mathcal{C}, \mathcal{D})$ is defined as $\mathcal{I}(\mathcal{C}, \mathcal{D}) = \mathcal{H}(\mathcal{C}) - \mathcal{H}(\mathcal{C}|\mathcal{D})$ and $\mathcal{H}(\cdot)$ is the entropy.

For unlabeled networks, we use normalized *goodness-of-fit* as the evaluation metric. *goodness-of-fit* can measure how well the representation of roles and the relations among these roles fit a given network (Wasserman and Faust 1994). In *goodness-of-fit*, it is assumed that the output of a role discovery method is an optimal model, and nodes belonging to the same role are predicted to be perfectly structurally equivalent. In real-world social networks, nodes belonging to the same role are only approximately structurally equivalent. The essence of *goodness-of-fit indices* is to measure how just how approximate are the approximate structural equivalences. If the optimal model holds, then all nodes belonging to the same role are exactly structurally equivalent.

In detail, given a social network with n vertices $V = \{v_1, v_2, \ldots, v_n\}$ and m roles, we have the adjacency matrix $A = \{A_{ij} \in \{0, 1\} | 1 \leq i, j \leq n\}$ and the role set $R = \{R_1, R_2, \ldots, R_m\}$, where $v_i \in R_j$ indicates node v_i belongs to the jth role, as obtained using DyNMF. Note that R partitions V, in the sense that each $v \in V$ belongs to exactly one $R_i \in R$. Then the density matrix Δ is defined as:

$$\Delta_{ij} = \begin{cases} \sum_{v_k \in R_i, v_l \in R_j} A_{kl} / (|R_i| \cdot |R_j|), & \text{if } i \neq j \\ \sum_{v_k \in R_i, v_l \in R_j} A_{kl} / (|R_i| \cdot (|R_j| - 1)), & \text{if } i = j. \end{cases} \tag{13}$$

Table 3 NMI for node clustering in air-traffic networks using different NE methods

	Brazil-air	Europe-air	USA-air
DeepWalk (Perozzi et al. 2014)	0.1303	0.0458	0.0766
LINE (Tang et al. 2015b)	0.2215	0.1563	0.1275
node2vec (Grover and Leskovec 2016)	0.2516	0.1722	0.0945
EP (Duran and Niepert 2017)	0.2283	0.1405	0.1007
graph2gauss (Vilnis and McCallum 2014)	0.1204	0.1109	0.0896
struc2vec (Ribeiro et al. 2017)	0.3758	0.2729	0.2486
DRNE (Tu et al. 2018)	0.5244	0.2766	0.2918
GraphWave (Donnat et al. 2018)	0.5040	0.3230	0.2452
*s2g*_el_d	0.5615	0.3234	0.3188
*s2g*_el_s	0.5396	0.2974	0.2967
*s2g*_kl_d	0.5527	0.3145	0.3212
*s2g*_kl_s	**0.5675**	**0.3280**	**0.3217**

In *struc2gauss*, el and kl mean expected likelihood and KL divergence, respectively. d and s mean diagonal and spherical covariances, respectively. The highest values are in bold

We also define block matrix B based on the discovered roles. In fact, there are several criteria which can be used to build the block matrix including perfect fit, zeroblock, oneblock and α density criterion (Wasserman and Faust 1994). Since real social network data rarely contain perfectly structural equivalent nodes (Faust and Wasserman 1992), perfect fit, zeroblock and oneblock criteria would not work well in real-world data and we use α density criterion to construct the block matrix B:

$$B_{ij} = \begin{cases} 0, & \text{if } \Delta_{ij} < \alpha \\ 1, & \text{if } \Delta_{ij} \geq \alpha \end{cases} \tag{14}$$

where α is the threshold to determine the values in blocks. α density criterion is based on the density of edges between nodes belong to the same role and defined as

$$\alpha = \sum_{1 \leq i, j \leq n} A_{ij}/(n(n-1)). \tag{15}$$

Based on the definitions of density matrix Δ and block matrix B, the *goodness-of-fit index* e is defined as

$$e = \sum_{1 \leq i, j \leq m} |B_{ij} - \Delta_{ij}|. \tag{16}$$

To make the evaluation metric value in the range of $[0, 1]$, we normalize *goodness-of-fit* by dividing r^2 where r is number of groups/roles. For more details about *goodness-of-fit indices*, please refer to Wasserman and Faust (1994).

Results The NMI values for node clustering on networks with labels are shown in Table 3 and the normalized *goodness-of-fit* values for networks without labels are

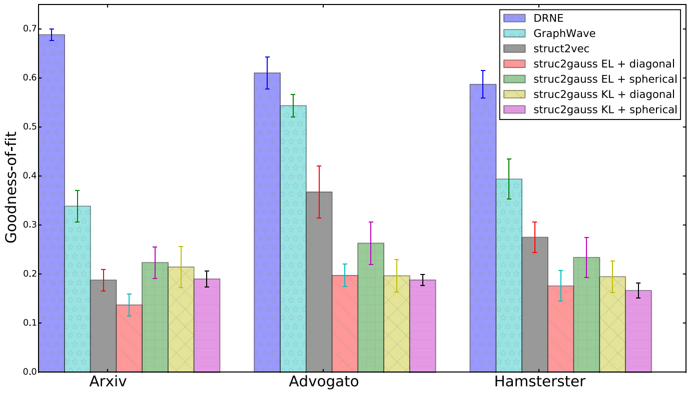


Fig. 4 Goodness-of-fit of global structure preserving embedding baselines and *struc2gauss* with different strategies on three real-world networks. Lower value means better performance

shown in Fig. 4. Note that random walk and neighbor based embedding methods, including DeepWalk, LINE, *node2vec*, EP and *graph2gauss*, aim at capturing local structural information and so are incapable of preserving structural roles. Hence, for simplicity, we will not compare them to these role preserving methods on networks without clustering labels.

From these results, some conclusions can be drawn:

- For both types of networks with and without clustering labels, *struc2gauss* outperforms all other methods in different evaluation metrics. It indicates the effectiveness of *struc2gauss* in capturing the global structural information.
- Comparing *struc2gauss* with diagonal and spherical covariances, it can be observed that spherical covariance can achieve better performance in node clustering. This finding is similar to the results of word embedding in Vilnis and McCallum (2014). A possible explanation could be: spherical covariance requires the diagonal elements to be the same which limits the representation power of covariance matrices but on the contrast enhance the representation power of the learned means. Since we only use means to represent nodes, the method with spherical covariance matrix could learn more relaxed means which leads to better performance.
- For baselines, *struc2vec*, GraphWave and DRNE can capture the structural role information to some extent since their performance is better than these random walk based methods, i.e., DeepWalk and *node2vec*, and neighbor-based method, i.e., EP and *graph2gauss*, while all of them fail in capturing the global structural information for node clustering.

5.4 Structural role classification

Node classification is another widely used task for embedding evaluation. Different from previous studies which focused on community structures, our approach aims to preserve the global role structures. Thus, we evaluate the effectiveness of *struc2gauss*

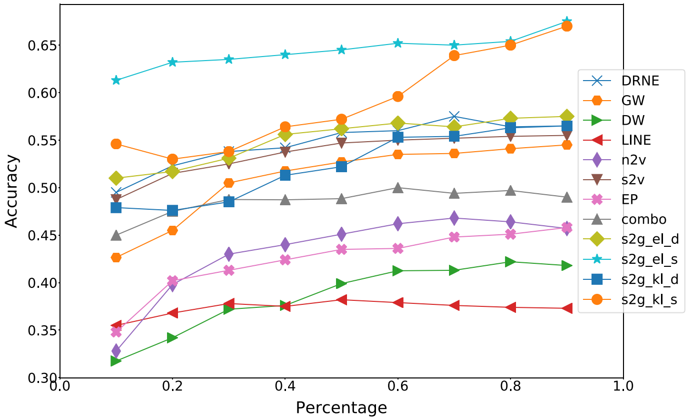


Fig. 5 Average accuracy for structural role classification in Europe-air network

in role classification task. Same to the node clustering task in Sect. 5.3, we use the latent representations learned by different methods as features. Each dataset is separated into training set and test set (we will explore the classification performance with different percentages of training set). To focus on the learned representation, we use logistic regression as the classifier.

Structural role classification as a supervised task, the ground-truth labels are required. Thus we only use two air-traffic networks for evaluation. We compare our approach to the same state-of-the-art NE algorithms as baselines used in Sect. 5.3, i.e., DeepWalk, LINE, *node2vec*, EP, *graph2gauss*, *struc2vec*, GraphWave and DRNE. Same to Tu et al. (2018), we also compare to four centrality measures, i.e., closeness centrality, betweenness centrality, eigenvector centrality and k-core. Since the combination of these four measures perform best (Tu et al. 2018), we only compare the classification performance of the combination as features in this task. The parameters of baselines and *struc2gauss*, we use the same settings in Sect. 5.3.

The average accuracies for structural role classification in Europe-air and USA-air are shown in Figs. 5 and 6. From the results, we can observe that:

- *struc2gauss* outperforms almost all other methods in both networks except DRNE in Europe-air network. In Europe-air network, *struc2gauss* with expected likelihood and spherical covariances, i.e., s2g_el_s, performs best. *struc2gauss* with KL divergence and spherical covariances, i.e., s2g_kl_s, achieves the second best performance especially when the training ratio is larger than 0.7. *struc2gauss* with diagonal covariances, i.e., s2g_el_d and s2g_kl_d, are on par with GraphWave, DRNE and *struc2vec* and outperform other methods. In the USA-air network, *struc2gauss* with different settings outperforms all baselines. This indicates the effectiveness of *struc2gauss* in modeling the structural role information. Although not the same combination of energy function and covariance form performs best in two networks, different variants of *struc2gauss* are always the best.
- Among the baselines, only *struc2vec*, GraphWave and DRNE can capture the structural role information so that they achieve better classification accuracy than other

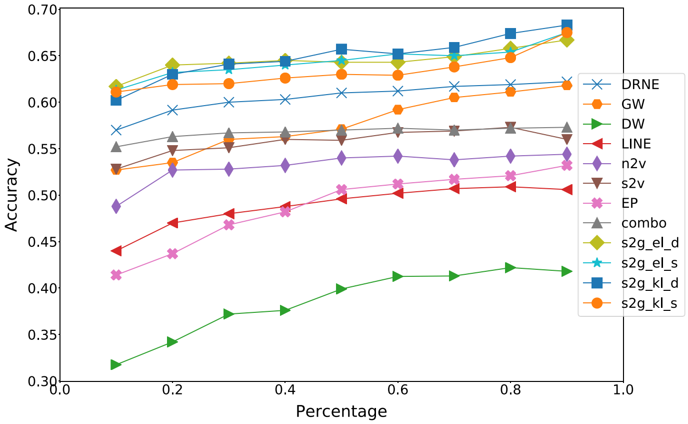


Fig. 6 Average accuracy for structural role classification in USA-air network

baselines. DRNE performs the best among these baselines since it captures regular equivalence. GraphWave and *struc2vec* are the second best baselines because they also aim to capture structural roles.

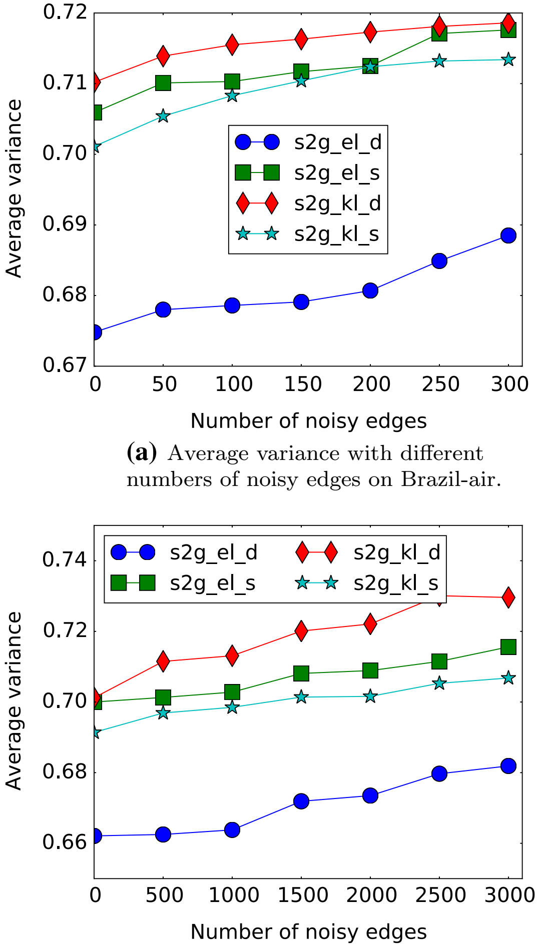
- Random walk and neighbor based NE methods only capture local community structures so they perform worse than *struc2gauss*, GraphWave, DRNE and our proposed *struc2gauss*. Node that methods such as DeepWalk, LINE and *node2vec*, although considering the first-, second- and/or higher-order proximity, still are not capable of modeling structural role information.

5.5 Uncertainty modeling

Mapping a node in a network into a distribution rather than a point vector allows us to model the uncertainty of the learned representation which is another advantage of *struc2gauss*. Different factors can lead to uncertainties of data. It is intuitive that the more noisy edges a node has, the less discriminative information it contains, thus making its embedding more uncertain. Similarly, incompleteness of information in the network can also bring uncertainties to the representation learning. Therefore, in this section, we study two factors: *noisy information* and *incomplete information*.

To verify these hypotheses, we conduct the following experiment using Brazil-air and Europe-air networks. For *noisy information*, we randomly insert certain number of edges to the network and then learn the latent representations and covariances. The average variance is used to measure the uncertainties. For Brazil-air network, we range the number of noisy edges from 50 to 300 and for Europe-air it ranges from 500 to 3000. For *incomplete information*, we randomly delete certain number of edges to the network to make it incomplete and then learn the latent representations and covariances. Similarly, for Brazil-air network, we range the number of removed edges from 50 to 300 and for Europe-air it ranges from 500 to 3000. The other parameter settings are same to Sect. 5.3.

Fig. 7 Uncertainties of embeddings with different levels of noise

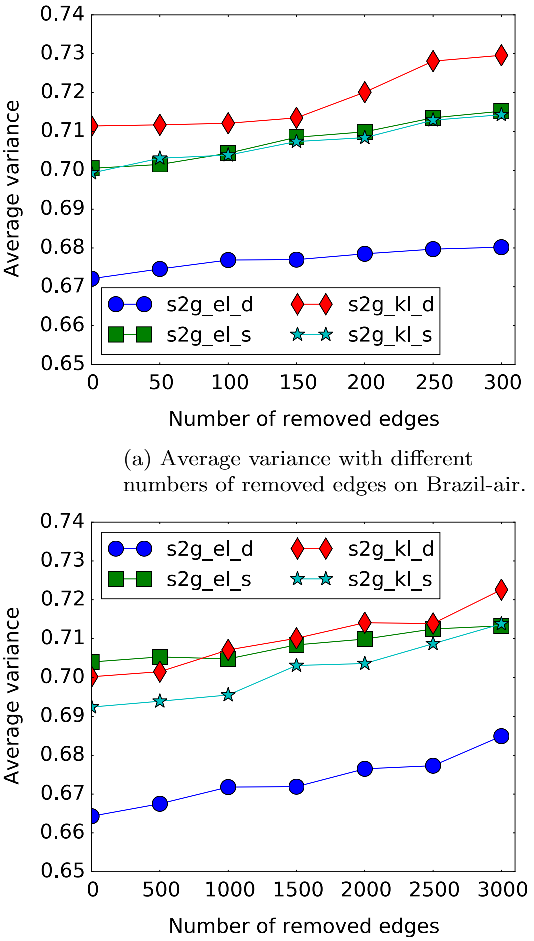


(a) Average variance with different numbers of noisy edges on Brazil-air.

(b) Average variance with different numbers of noisy edges on Europe-air.

The results are shown in Figs. 7 and 8. It can be observed that (1) with more noisy edges being added to the networks and (2) with more removed edges from the networks, average variance values become larger. *struc2gauss* with different energy functions and covariance forms have the same trend. This demonstrates that our proposed *struc2gauss* is able to model the uncertainties of learned node representations. It is interesting that *struc2gauss* with expected likelihood and diagonal covariance (s2g_el_d) always has the lowest average variance while *struc2gauss* with KL divergence and diagonal (s2g_kl_d) always has the largest value. This may result from the learning mechanism of different energy functions when measuring the distance between two distributions. To clarify the results, we also list the NMI for the clustering task in Tables 4 and 5. Compared to the original Gaussian embedding method, we

Fig. 8 Uncertainties of embeddings with different levels of incompleteness



(a) Average variance with different numbers of removed edges on Brazil-air.

(b) Average variance with different numbers of removed edges on Europe-air.

again show the effectiveness of our method in preserving structural role and modeling uncertainties.

5.6 Influence of similarity measures

As we mentioned not all structural similarity measures can capture the global structural role information, to validate the rationale to select RoleSim as the similarity measure for structural role information, we investigate the influence of different similarity measures on learning node representations. In specific, we select two other widely used structural similarity measures, i.e., SimRank (Jeh and Widom 2002) and MatchSim (Lin et al. 2009), and we incorporate these measures by replacing RoleSim in our framework. The datasets and evaluation metrics used in this experiment are

Table 4 NMI for node clustering in Brazil-air network with different numbers of noisy edges

# noisy edges	0	50	100	150	200	250	300
graph2gauss	0.1204	0.1032	0.0903	0.0913	0.0852	0.0833	0.0683
*s2g*_el_d	0.5615	0.5165	0.5161	0.5122	0.4810	0.4754	0.4787
*s2g*_el_s	0.5396	0.4338	0.4180	0.4152	0.4102	0.3956	0.3924
*s2g*_kl_d	0.5527	0.5186	0.5036	0.4940	0.4824	0.4736	0.4103
*s2g*_kl_s	0.5527	0.5310	0.5214	0.4951	0.4895	0.4621	0.4651

Table 5 NMI for node clustering in Europe-air network with different numbers of noisy edges

# noisy edges	0	500	1000	1500	2000	2500	3000
graph2gauss	0.1109	0.0776	0.0727	0.0716	0.0634	0.0702	0.0613
*s2g*_el_d	0.3234	0.1767	0.1634	0.1694	0.1492	0.1431	0.1413
*s2g*_el_s	0.2974	0.1613	0.1505	0.1432	0.1452	0.1285	0.1042
*s2g*_kl_d	0.3145	0.2664	0.2014	0.1854	0.1802	0.1634	0.1361
*s2g*_kl_s	0.3280	0.3024	0.2930	0.1504	0.1514	0.1414	0.1367

the same to Sect. 5.3. For simplicity, we only show the results of *struc2gauss* using KL divergence with spherical covariance in this experiment because different variants perform similarly in previous experiments.

The NMI values for networks with labels are shown in Tables 6 and the *goodness-of-fit* values are shown in Fig. 9. We can come to the following conclusions:

- RoleSim outperforms other two similarity measures in both types of networks with and without clustering labels. It indicates RoleSim can better capture the global structural information. Performance of MatchSim varies on different networks and is similar to *struc2vec*. Thus, it can capture the global structural information to some extent.
- SimRank performs worse than other similarity measures as well as *struc2vec* (Table 3). Considering the basic assumption of SimRank that “two objects are similar if they relate to similar objects”, it computes the similarity also via relations between nodes so that the mechanism is similar to random walk based methods which have been proved not being capable of capturing the global structural information (Lyu et al. 2017).

5.7 Parameter sensitivity

We consider two types of parameters in *struc2gauss*: (1) parameters also used in other NE methods including *latent dimensions*, *number of samples per node* and *number of positive/negative nodes per node*; and (2) parameters only used in Gaussian embedding including mean constraint C and covariance constraint c_{max} (note that we fix the minimal covariance c_{min} to be 0.5 for simplicity). In order to evaluate how changes to these parameters affect performance, we conducted the same node

Table 6 NMI for node clustering in air-traffic networks of Brazil, Europe and USA using *struc2gauss* with different similarity measures

	Brazil-air	Europe-air	USA-air
SimRank	0.1695 ± 0.013	0.0524 ± 0.009	0.0887 ± 0.025
MatchSim	0.3534 ± 0.024	0.2389 ± 0.020	0.0913 ± 0.013
RoleSim	**0.5675** ± **0.032**	**0.3280** ± **0.019**	**0.3217** ± **0.023**

Results in bold represent the best performance

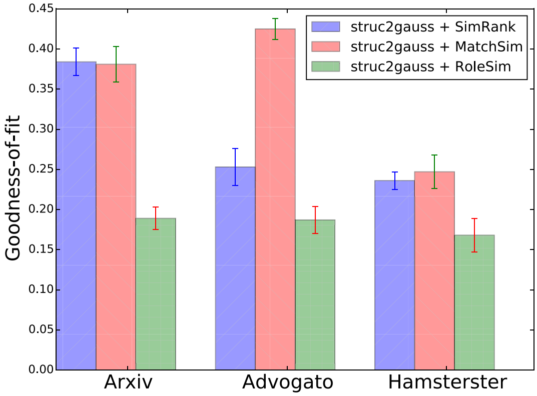
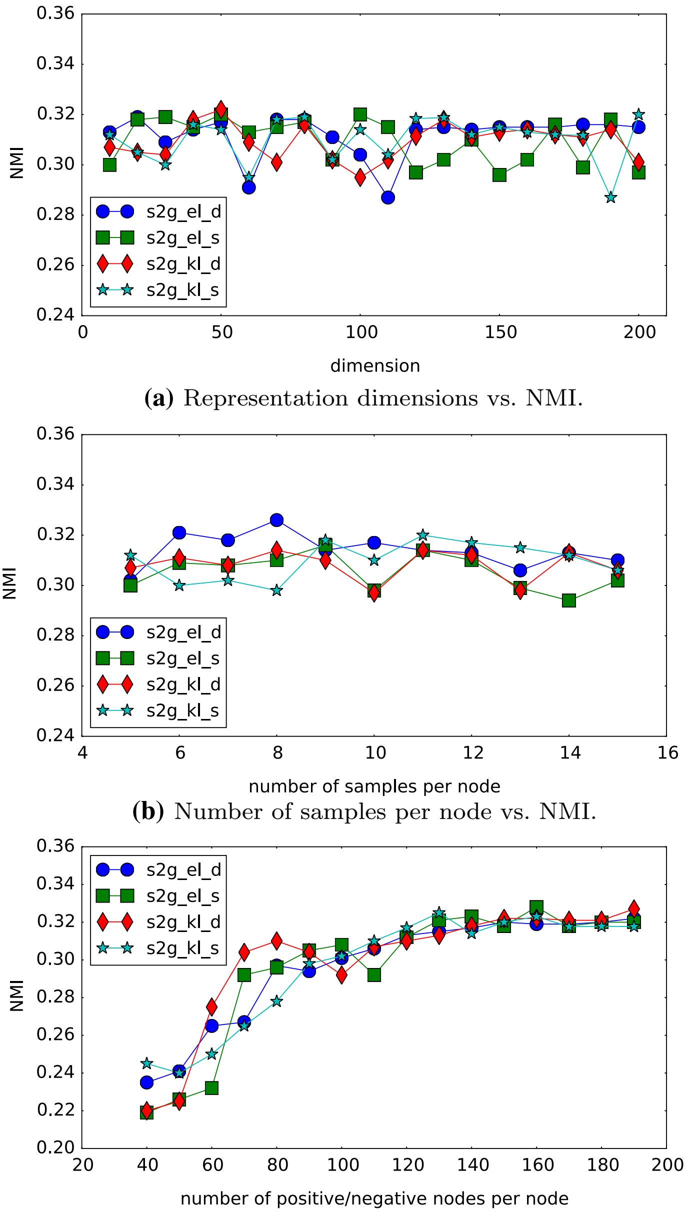


Fig. 9 Goodness-of-fit of *struc2gauss* with different similarity measures. Lower values are better

clustering experiment on the labeled USA-air network introduced in Sect. 5.3. In the interest of brevity, we tune one parameter by fixing all other parameters. In specific, the number of latent dimensions varies from 10 to 200, the number of samples varies from 5 to 15 and the number of positive/negative nodes varies from 40 to 190. Mean constraint C is from 1 to 10, and covariance constraint c_{max} ranges from 1 to 10.

The results of parameter sensitivity are shown in Figs. 10 and 11. It can be observed from Fig. 10a, b that the trends are relatively stable, i.e., the performance is insensitive to the changes of representation dimensions and numbers of samples. The performance of clustering is improved with the increase of numbers of positive/negative nodes shown in Fig. 10c. Therefore, we can conclude that *struc2guass* is more stable than other methods. It has been reported that other methods, e.g., DeepWalk (Perozzi et al. 2014), LINE (Tang et al. 2015b) and *node2vec* (Grover and Leskovec 2016), are sensitive to many parameters. In general, more dimensions, more walks and more context can achieve better performance. However, it is difficult to search for the best combination of parameters in practice and it may also lead to overfitting. For Gaussian embedding specific parameters C and c_{max}, both trends are stable, i.e., the selection of these contraints have little effect on the performance. Although with larger mean constraint C, the NMI decreases but the difference is not huge.

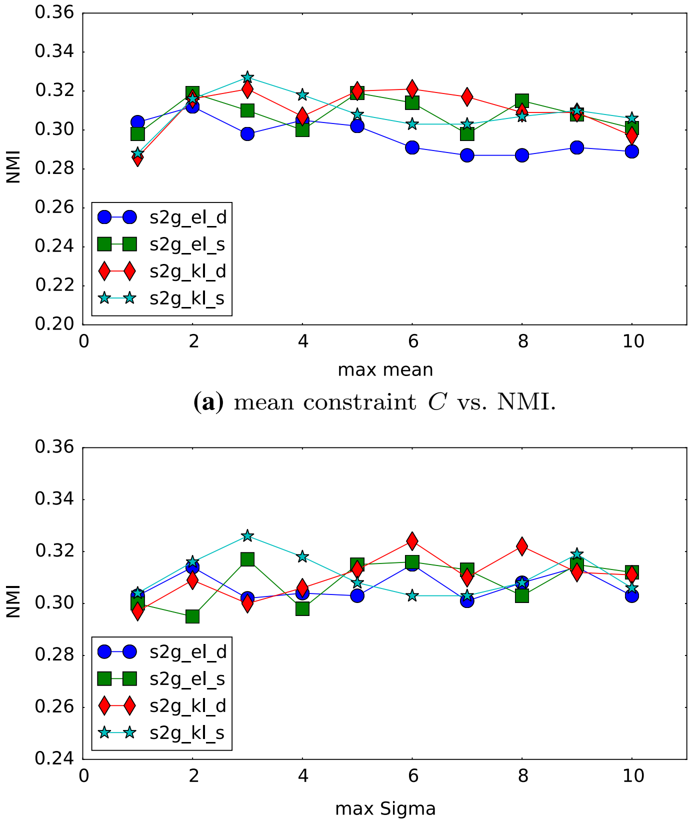


(a) Representation dimensions vs. NMI.

(b) Number of samples per node vs. NMI.

(c) Number of positive/negative nodes per node vs. NMI.

Fig. 10 Parameter sensitivity study



(a) mean constraint C vs. NMI.

(b) covariance constraint c_{max} vs. NMI.

Fig. 11 Parameter sensitivity in Gaussian distributions

5.8 Efficiency and effectiveness study

As discussed above in Sect. 4.5, the high computational complexity is one of the major issues in our method. In this experiment, we empirically study this computational issue by comparing the run-time and performance of different global structural preserving baselines and a heuristic method to accelerate the RoleSim measures. The heuristic method, named Fast *struc2gauss*, is introduced in Sect. 4.5: we set the similarity to be 0 if two nodes have a large difference in degrees to avoid more computing for dissimilar node pairs. For simplicity, we only test *struc2gauss* with KL and spherical covariance. Also, we only consider embedding methods that can preserve the structural role information as baselines, i.e., GraphWave, *struc2vec* and DRNE.

We conduct the experiments on the larger networks without ground-truth labels because on smaller networks the run-time differences are not significant. The run-time comparison is shown in Table 7 and the performance comparison is shown in Table 8. Note that NA in these tables because these methods reported a memory error and did

Table 7 Run-time for different structural role preserving network embedding methods

	GraphWAVE	*struc2vec*	DRNE	*struc2gauss*	Fast *struc2gauss*
Arxiv	90.68s	10+h	159.43s	2h	886.93s
Advogato	172.13s	10+h	191.52s	4h	1962.68s
Hamsterster	24.25s	10+h	85.93s	1h	456.24s
Anybeat	NA	NA	1094.64s	13h	5h
Epinion	NA	NA	2938.83s	20h+	12h

Table 8 Performance (*goodness-of-fit*) of different structural role preserving network embedding methods

	GraphWAVE	*struc2vec*	DRNE	*struc2gauss*	Fast *struc2gauss*
Arxiv	0.5435	0.3674	0.6822	0.1880	0.1983
Advogato	0.3938	0.2751	0.6102	0.1852	0.2012
Hamsterster	0.3385	0.1878	0.5939	0.1666	0.1790
Anybeat	NA	NA	0.5639	0.1597	0.1622
Epinion	NA	NA	0.4978	0.2270	0.2452

not obtain any results. To make a fair comparison, all these methods are run in the same machine with 128GB memory and GPU have not been used for DRNE. From these results, it can be observed: (1) although the computational issue still exists, our method can achieve good performance compared to state-of-the-art structural role preserving network embedding methods such as GraphWAVE and *struc2vec*. (2) Although DRNE is much fast, its performance is worse than our method and other baselines. Moreover, it is incapable of modeling uncertainties. (3) Fast *struc2gauss* can effectively accelerate RoleSim computing and achieve comparable performance in role clustering.

6 Discussion

The proposed *struc2gauss* is a flexible framework for node representations. As shown in Fig. 2, different similarity measures can be incorporated into this framework and empirical studies will be presented in Sect. 5.6. Furthermore, other types of methods which model structural information can be utilized in *struc2gauss* as well.

To illustrate the potential to incorporate different methods, we categorize different methods for capturing structural information into three types:

- **Similarity-based methods**. Similarity-based methods calculate pairwise similarity based on the structural information of a given network. Related work has been reviewed in Sect. 2.2.
- **Ranking-based methods**. PageRank (Page et al. 1999) and HITS (Kleinberg 1999) are two most representative ranking-based methods which learns the structural information. PageRank has been used for NE in (Ma et al. 2017).

- **Partition-based methods**. This type of methods, e.g., role discovery, aims to partition nodes into disjoint or overlapping groups, e.g., REGE (Borgatti and Everett 1993) and RolX (Henderson et al. 2012).

In this paper, we focus on **similarity-based methods**. For **ranking-based methods**, we can use a fixed sliding window on the ranking list, then given a node the nodes within the window can be viewed as the context. In fact, this mechanism is similar to DeepWalk. For **partition-based methods**, we can consider the nodes in the same group as the context for each other.

7 Conclusions and future work

Two major limitations exist in previous NE studies: i.e., **structure preservation** and **uncertainty modeling**. Random-walk based NE methods fail in capturing global structural information and representing a node into a point vector are not capable of modeling the uncertainties of node representations.

We proposed a flexible structure preserving network embedding framework, *struc2gauss*, to tackle these limitations. On the one hand, *struc2gauss* learns node representations based on structural similarity measures so that global structural information can be taken into consideration. On the other hand, *struc2gauss* utilizes Gaussian embedding to represent each node as a Gaussian distribution where the mean indicates the position of this node in the embedding space and the covariance represents its uncertainty.

We experimentally compared three different structural similarity measures for networks and two different energy functions for Gaussian embedding. By conducting experiments from different perspectives, we demonstrated that *struc2gauss* excels in capturing global structural information, compared to state-of-the-art NE techniques such as DeepWalk, *node2vec* and *struc2vec*. It outperforms other competitor methods in role discovery task and structural role classification on several real-world networks. It also overcomes the limitation of uncertainty modeling and is capable of capturing different levels of uncertainties. Additionally, *struc2gauss* is less sensitive to different parameters which makes it more stable in practice without putting more effort in tuning parameters.

In the future, we will explore faster RoleSim measures for more scalable NE methods, for example, fast method to select k most similar nodes for a given node. Also, it is a promising research direction to investigate different strategies to model global structural information except structural similarity in NE tasks. Besides, other future investigations in this area include learning node representations in dynamic and temporal networks.

References

Airoldi EM, Blei DM, Fienberg SE, Xing EP (2008) Mixed membership stochastic blockmodels. J Mach Learn Res 9(Sep):1981–2014

Antonellis I, Molina HG, Chang CC (2008) Simrank++: query rewriting through link analysis of the click graph. Proc VLDB Endow 1(1):408–421

Ba JL, Kiros JR, Hinton GE (2016) Layer normalization. arXiv preprint arXiv:1607.06450

Belkin M, Niyogi P (2001) Laplacian eigenmaps and spectral techniques for embedding and clustering. In: Advances in neural information processing systems. pp 585–591

Bojchevski A, Günnemann S (2017) Deep gaussian embedding of attributed graphs: unsupervised inductive learning via ranking. arXiv preprint arXiv:1707.03815

Borgatti SP, Everett MG (1993) Two algorithms for computing regular equivalence. Soc Netw 15(4):361–376

Cai H, Zheng VW, Chang KCC (2018) A comprehensive survey of graph embedding: problems, techniques, and applications. IEEE Trans Knowl Data Eng 30(9):1616–1637

Cao S, Lu W, Xu Q (2015) Grarep: Learning graph representations with global structural information. In: Proceedings of the 24th ACM international on conference on information and knowledge management. ACM, pp 891–900

Chang S, Han W, Tang J, Qi GJ, Aggarwal CC, Huang TS (2015) Heterogeneous network embedding via deep architectures. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 119–128

Cui P, Wang X, Pei J, Zhu W (2018) A survey on network embedding. IEEE Trans Knowl Data Eng 31(5):833–852

Donnat C, Zitnik M, Hallac D, Leskovec J (2018) Learning structural node embeddings via diffusion wavelets. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 1320–1329

Dos Santos L, Piwowarski B, Gallinari P (2016) Multilabel classification on heterogeneous graphs with gaussian embeddings. In: Joint European conference on machine learning and knowledge discovery in databases. Springer, pp 606–622

Duchi J, Hazan E, Singer Y (2011) Adaptive subgradient methods for online learning and stochastic optimization. J Mach Learn Res 12(Jul):2121–2159

Duran AG, Niepert M (2017) Learning graph representations with embedding propagation. In: Advances in neural information processing systems. pp 5119–5130

Faust K, Wasserman S (1992) Blockmodels: interpretation and evaluation. Soc Netw 14(1):5–61

Fortunato S (2010) Community detection in graphs. Phys Rep 486(3):75–174

Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE (2017) Neural message passing for quantum chemistry. In: International conference on machine learning. pp 1263–1272

Goyal P, Ferrara E (2018) Graph embedding techniques, applications, and performance: a survey. Knowl Based Syst 151:78–94

Grover A, Leskovec J (2016) node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 855–864

Hamilton W, Ying Z, Leskovec J (2017a) Inductive representation learning on large graphs. In: Advances in neural information processing systems. pp 1024–1034

Hamilton WL, Ying R, Leskovec J (2017b) Representation learning on graphs: methods and applications. IEEE Data Eng Bull 40(3):52–74

He S, Liu K, Ji G, Zhao J (2015) Learning to represent knowledge graphs with gaussian embedding. In: Proceedings of the 24th ACM international on conference on information and knowledge management. ACM, pp 623–632

Heimann M, Shen H, Safavi T, Koutra D (2018) Regal: Representation learning-based graph alignment. In: Proceedings of the 27th ACM international conference on information and knowledge management. pp 117–126

Henderson K, Gallagher B, Li L, Akoglu L, Eliassi-Rad T, Tong H, Faloutsos C (2011) It's who you know: graph mining using recursive structural features. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 663–671

Henderson K, Gallagher B, Eliassi-Rad T, Tong H, Basu S, Akoglu L, Koutra D, Faloutsos C, Li L (2012) Rolx: structural role extraction and mining in large graphs. In: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 1231–1239

Jeh G, Widom J (2002) Simrank: a measure of structural-context similarity. In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 538–543

Jin R, Lee VE, Hong H (2011) Axiomatic ranking of network role similarity. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 922–930

Jin R, Lee VE, Li L (2014) Scalable and axiomatic ranking of network role similarity. ACM Trans Knowl Discov Data (TKDD) 8(1):3

Kipf TN, Welling M (2016) Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907

Kleinberg JM (1999) Authoritative sources in a hyperlinked environment. J ACM (JACM) 46(5):604–632

Li J, Dani H, Hu X, Tang J, Chang Y, Liu H (2017) Attributed network embedding for learning in a dynamic environment. In: Proceedings of the 2017 ACM on conference on information and knowledge management. pp 387–396

Liben-Nowell D, Kleinberg J (2007) The link-prediction problem for social networks. J Assoc Inf Sci Technol 58(7):1019–1031

Lin Z, Lyu MR, King I (2009) Matchsim: a novel neighbor-based similarity measure with maximum neighborhood matching. In: Proceedings of the 18th ACM conference on Information and knowledge management. ACM, pp 1613–1616

Lyu T, Zhang Y, Zhang Y (2017) Enhancing the network embedding quality with structural similarity. In: Proceedings of the 2017 ACM on conference on information and knowledge management. ACM, pp 147–156

Ma Y, Wang S, Ren Z, Yin D, Tang J (2017) Preserving local and global information for network embedding. arXiv preprint arXiv:1710.07266

Mikolov T, Chen K, Corrado G, Dean J (2013a) Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781

Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J (2013b) Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp 3111–3119

Page L, Brin S, Motwani R, Winograd T (1999) The pagerank citation ranking: Bringing order to the web. Tech. rep, Stanford InfoLab

Pei Y, Zhang J, Fletcher G, Pechenizkiy M (2018) Dynmf: role analytics in dynamic social networks. In: Proceedings of the 27th international joint conference on artificial intelligence. AAAI Press, pp 3818–3824

Pennington J, Socher R, Manning C (2014) Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp 1532–1543

Perozzi B, Al-Rfou R, Skiena S (2014) Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 701–710

Perozzi B, Kulkarni V, Skiena S (2016) Walklets: Multiscale graph embeddings for interpretable network classification. arXiv preprint arXiv:1605.02115

Ribeiro LF, Saverese PH, Figueiredo DR (2017) struc2vec: Learning node representations from structural identity. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 385–394

Rossi RA, Ahmed NK (2015) Role discovery in networks. IEEE Trans Knowl Data Eng 27(4):1112–1131

Scarselli F, Gori M, Tsoi AC, Hagenbuchner M, Monfardini G (2008) The graph neural network model. IEEE Trans Neural Netw 20(1):61–80

Shaw B, Jebara T (2009) Structure preserving embedding. In: Proceedings of the 26th annual international conference on machine learning. ACM, pp 937–944

Tang J, Qu M, Mei Q (2015a) Pte: Predictive text embedding through large-scale heterogeneous text networks. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 1165–1174

Tang J, Qu M, Wang M, Zhang M, Yan J, Mei Q (2015b) Line: Large-scale information network embedding. In: Proceedings of the 24th international conference on World Wide Web, international World Wide Web conferences steering committee. pp 1067–1077

Tenenbaum JB, De Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science 290(5500):2319–2323

Tu K, Cui P, Wang X, Yu PS, Zhu W (2018) Deep recursive network embedding with regular equivalence. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 2357–2366

Velickovic P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y (2017) Graph attention networks. arXiv preprint arXiv:1710.10903

Vilnis L, McCallum A (2014) Word representations via gaussian embedding. arXiv preprint arXiv:1412.6623

Wang D, Cui P, Zhu W (2016) Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 1225–1234

Wang S, Aggarwal C, Tang J, Liu H (2017) Attributed signed network embedding. In: Proceedings of the 2017 ACM on conference on information and knowledge management. ACM, pp 137–146

Wasserman S, Faust K (1994) Social network analysis: methods and applications, vol 8. Cambridge University Press, Cambridge

Zhao P, Han J, Sun Y (2009) P-rank: a comprehensive structural similarity measure over information networks. In: Proceedings of the 18th ACM conference on information and knowledge management. ACM, pp 553–562

Zhu D, Cui P, Wang D, Zhu W (2018) Deep variational network embedding in wasserstein space. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 2827–2836

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

Yulong Pei[1] · Xin Du[1] · Jianpeng Zhang[2] · George Fletcher[1] · Mykola Pechenizkiy[1]

Xin Du
x.du@tue.nl

Jianpeng Zhang
zjp@ndsc.com.cn

George Fletcher
g.h.l.fletcher@tue.nl

Mykola Pechenizkiy
m.pechenizkiy@tue.nl

[1] Department of Mathematics and Computer Science, Eindhoven University of Technology, 5600 MB Eindhoven, The Netherlands

[2] National Digital Switching System Engineering Technology R&D Center, Zhengzhou, China