\textit{struc2gauss:} Structure Preserving Network Embedding via Gaussian Embedding

Yulong Pei · Xin Du · Jianpeng Zhang · George Fletcher · Mykola Pechenizkiy

Received: date / Accepted: date

\textbf{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: \textit{structure preservation} and \textit{uncertainty modeling}. Almost all previous methods represent a node into a point in space and focus on the local structural information, i.e., neighborhood information. However, neighborhood information does not capture the global structural information and point vector representation fails in modeling the uncertainty of node representations. In this paper, we propose a new NE framework, \textit{struc2gauss}, which learns node representations in the space of Gaussian distributions and performs network embedding based on global structural information. \textit{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 both synthetic and real-world data sets demonstrate that \textit{struc2gauss} effectively captures the global structural information while state-of-the-art network embedding methods fails to, outperforms other meth-
ods on the structure-based clustering task and provides more information on uncertainties of node representations.

**Keywords** Gaussian Embedding · Structural Similarity · Node Representations

1 Introduction

Network analysis consists of numerous tasks including community detection [9], role discovery [31], link prediction [20], 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 [5, 10, 29, 30].

Previous NE techniques mainly relied on eigendecomposition [32, 35], 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 [24, 25] and GloVe [28]. By drawing an analogy between random walks on networks and word sequences in text, DeepWalk [29] 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 [5, 10, 34] or incorporating more information for node representations such as attributes [19, 37] and heterogeneity [6, 33].

Although a variety of NE methods have been proposed, two major limitations exist in previous NE studies: (1) **Structure preservation.** Previous studies applied random walk to learn representations. However, random walk based embedding strategies can only capture local structural information, i.e., first-order and higher-order proximity within the neighborhood of the target node [22] and fail in capturing the global structural information, e.g., structural or regular equivalence [38]. An example of global structural information and local structural information is shown in Fig. 1 and empirical evidence based on this example for illustrating this limitation will be shown in Section 5.1. (2) **Uncertainty modeling.** Previous methods represent a node into a point vector in the learned embedding space. However, real-world networks may be noisy and imbalanced. Point vector representations are deterministic [7] and are not capable of modeling the uncertainties of node representations.

There are limited studies trying to address these limitations in the literature. For instance, **struc2vec** [30] builds a hierarchy to measure similarity at different scales, and constructs a multilayer graph to encode the structural similarities. SNS [22] discovers graphlets as a pre-processing step to obtain the
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 regular equivalence and (2) two groups (groups are shown by the dashed ellipses) based on local structural information, i.e., the community. 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.

structural similar nodes. However, both studies aim only to solve the problem of structure preservation to some extent. Thus the limitation of uncertainty modeling remains a challenge. [7] and [3] 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. Therefore, the problem of structure preservation has not been solved in these studies.

In this paper, we propose struc2gauss, a new structure 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 structural similarity measures 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 three different structural similarity measures for networks, i.e., RoleSim, MatchSim and SimRank, and two different energy functions for Gaussian embedding to calculating the closeness of two embedded Gaussian distributions, i.e., expected likelihood and KL divergence.

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 based on global structural information.
– We investigate the influence of different energy functions and different structural similarity measures on NE to preserve global structural information of networks.

– We conduct extensive experiments which demonstrate the effectiveness of struc2gauss in capturing the global structural 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 Section 3. Section 4 explains the technical details of struc2gauss. In Section 5 we then discuss our experimental study. Finally, in Section 6 we draw conclusions and outline directions for future work.

2 Related Work

2.1 Network Embedding

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. The learned node representations can boost the performance in many network analysis tasks, e.g., community detection and link prediction. Previous methods mainly focused on matrix factorization and eigendecomposition \cite{32, 35} to reduce the dimension of network data.

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 \cite{24, 25} learn low-rank representations of words in text based on word context and show promising results of different NLP tasks. Based on word2vec, DeepWalk \cite{29} 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 \cite{5, 10, 34} or incorporating more information for node representations such as attributes \cite{19, 37} and heterogeneity \cite{6, 33}. We refer the reader to \cite{11} for more details.

However, almost all these state-of-the-art methods only concern the local structural information represented by random walks and fail to capture global structural information. SNS \cite{22} and struc2vec are two exceptions which take global structural information into consideration. SNS uses graphlet information for structural similarity calculation as a pre-processing step and 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.
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 \cite{15} 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++ \cite{2} adds an evidence weight which partially compensates for the neighbor matching cardinality problem. P-Rank \cite{39} extends SimRank by jointly encoding both in- and out-link relationships into structural similarity computation. MatchSim \cite{21} uses maximal matching of neighbors to calculate the structural similarity. RoleSim \cite{16} 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 \cite{31}. In social science research, roles are represented as concepts of equivalence \cite{38}. 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 \cite{1} 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 \cite{13} employs ReFeX \cite{14} 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 \cite{9}. 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 and global structural information in Section 1 using the example in Fig. 1. In this study, we only consider the global structural information, so without mentioning it explicitly, structural information indicates the global one. We formally define the problem of structure preserving network embedding.

**Definition 1 Structure 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 **Structure Preserving Network Embedding** aims to represent each node $v \in V$ into a Gaussian distribution with mean $\mu$ and covariance $\Sigma$ in a low-dimensional space $\mathbb{R}^d$, i.e., learning a function $f : V \rightarrow \mathcal{N}(x; \mu, \Sigma)$, where $\mu \in \mathbb{R}^d$ is the mean, $\Sigma \in \mathbb{R}^{d \times d}$ is the covariance
Similarity measures:
- SimRank
- SimRank++
- MatchSim
- RoleSim
- …

Similarity Matrix

Gaussian Embedding
- EL Energy
- KL Energy

Training set sampling

Fig. 2: Overview of the struc2gauss framework.

and \( d \ll |V| \). In the space \( \mathbb{R}^d \), the global structural information of nodes can be preserved and the uncertainty of node representations can be captured.

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.

4.1 Structural Similarity Calculation

It has been theoretically proved that random walk sampling based NE methods are not capable of capturing structural equivalence [22]. Thus, to capture global structural information, we calculate the structural similarity as a preprocessing step similar to [22, 30].

In this paper, we use RoleSim [16] for the structural similarity since it satisfies all the requirements of Axiomatic Role Similarity Properties for modeling the equivalence [16]. RoleSim also generalizes Jaccard coefficient and corresponds linearly to the maximal weighted matching. RoleSim metric between two nodes \( u \) and \( v \) is defined as:

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

where \( N(u) \) and \( N(v) \) are 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 \( \beta \) is a decay factor where \( 0 < \beta < 1 \). 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 $k^{th}$ iteration $R^k$ scores for the $(k - 1)^{th}$ iteration’s values, $R^{k-1}$ using:
  \[ R^k(u, v) = (1 - \beta) \max_{M(u,v)} \frac{\sum_{(x,y) \in M(u,v)} R^{k-1}(x, y)}{N(u) + N(v) - |M(u,v)|} + \beta \]
- Repeat Step 2 until $R$ values converge for each pair of nodes.

Note that other ways to capture global structural information will be discussed in Section 4.6 and other structural similarity methods will be compared in Section 5.3 empirically.

4.2 Training Set Sampling

To learn node representations using Gaussian embedding, we have to sample training set based on the similarity matrix. For node $v$, we rank its similarity values towards other nodes and then select top-$k$ most similar nodes $u_i, i = 1, ..., k$ as its positive set $I_+ = \{(v, u_i) | i = 1, ..., k\}$. For the negative set, we randomly select the same number of nodes $\{u'_i, i = 1, ..., k\}$ same to [36], i.e., $I_- = \{(v, u'_i) | i = 1, ..., k\}$. 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.

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 [36]. Gaussian embedding has been utilized in different graph mining tasks including triplet classification on knowledge graphs [12], multi-label classification on heterogeneous graphs [7] and link prediction and node classification on attributed graphs [3].

Gaussian embedding trains with a ranking-based loss based on the ranks of positive and negative samples. Following [36], we choose the max-margin ranking objective which can push scores of positive pairs above negatives by a margin defined as:

\[ L = \sum_{(v,u) \in I_+} \sum_{(v',u') \in I_-} \max(0, m - \mathcal{E}(v,u) + \mathcal{E}(v',u')) \]
where $\Gamma^+$ and $\Gamma^-$ are the positive and negative pairs, respectively. $\mathcal{E}(\cdot, \cdot)$ is the energy function, $v$ and $u$ are the learned Gaussian distributions for two nodes 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.

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:

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

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

$$\mathcal{E}_{EL}(z_i, z_j) = \log E(z_i, z_j) = \log N(0; \mu_i - \mu_j, \Sigma_i + \Sigma_j) \quad (5)$$

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

$$\frac{\partial \mathcal{E}_{EL}(z_i, z_j)}{\partial \mu_i} = -\frac{\partial \mathcal{E}_{EL}(z_i, z_j)}{\partial \mu_j} = -\Delta_{ij} \quad (6)$$

where $\Delta_{ij} = (\Sigma_i + \Sigma_j)^{-1}(\mu_i - \mu_j)$ [12,36]. 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):

$$\mathcal{E}_{KL}(z_i, z_j) = D_{KL}(z_i, z_j) = \int_{x \in \mathbb{R}} N(x; \mu_i, \Sigma_i) \log \frac{N(x; \mu_j, \sigma_j)}{N(x; \mu_i, \Sigma_i)} dx \quad (7)$$

$$= \frac{1}{2} \left\{ \text{tr}(\Sigma_i^{-1} \Sigma_j) + (\mu_i - \mu_j)^T \Sigma_i^{-1}(\mu_i - \mu_j) - \log \frac{\det(\Sigma_j)}{\det(\Sigma_i)} - d \right\}$$

where $\text{tr}$ is the trace of a matrix.
where $d$ is the number of dimensions. Similarly, we can compute the gradients of this energy function with respect to the means $\mu$ and covariances $\Sigma$:

$$\frac{\partial E_{KL}(z_i, z_j)}{\partial \mu_i} = - \frac{\partial E_{KL}(z_i, z_j)}{\partial \mu_j} = -\Delta'_{ij}$$

$$\frac{\partial E_{KL}(z_i, z_j)}{\partial \Sigma_i} = \frac{1}{2}(\Sigma_i^{-1} \Sigma_j^{-1} + \Delta'_{ij} \Delta'_{ij}^T - \Sigma_i^{-1})$$

$$\frac{\partial E_{KL}(z_i, z_j)}{\partial \Sigma_j} = \frac{1}{2}(\Sigma_j^{-1} - \Sigma_i^{-1})$$

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:

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

4.4 Learning

To avoid overfitting, we regularize the means and covariances to learn the embedding. Due to the different geometric characteristics, two different hard constraint strategies have been used for means and covariances, respectively. In particular, we have

$$\|\mu_i\| \leq C, \ \forall i$$

$$c_{min} I \preceq \Sigma_i \preceq c_{max} I, \ \forall i.$$  \hspace{1cm} (10)

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 [8] 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.

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 $G$ [16].

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 $O(n \log k)$. 

Algorithm 1 The Learning Algorithm of \textit{struc2gauss}

\textbf{Input:} An energy function $E(z_i, z_j)$, a graph $G = (V, E)$, embedding dimension $d$, constraint values $c_{\text{max}}$ and $c_{\text{min}}$ for covariance, learning rate $\alpha$, and maximum epochs $n$.

\textbf{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. (6)
13: end if
14: Regularize $\mu$ and $\Sigma$ with constraint in Eq. (10) and (11)
15: end while

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.

4.6 Discussion

The proposed \textit{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 Section 5.3. Furthermore, other types of methods which model structural information can be utilized in \textit{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 Section 2.2

- **Ranking-based methods.** PageRank [26] and HITS [18] are two most representative ranking-based methods which learns the structural information. PageRank has been used for NE in [23].
– **Partition-based methods.** This type of methods, e.g., role discovery, aims to partition nodes into disjoint or overlapping groups, e.g., REGE [4] and RolX [13].

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.

## 5 Experiments

We evaluate *struc2gauss* in different scenarios 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.

### 5.1 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 in two ways: (1) based on global structural information they belong to three groups, i.e., \{0, 1, 2, 3\} (yellow color), \{4, 5, 6, 7\} (blue color) and \{8, 9\} (red color) and (2) based on local structural information they belong to two groups, i.e., \{0, 1, 4, 5, 6, 8\} and \{2, 3, 6, 7, 9\}. In this study, we only consider the global structural information. Note that from the perspective of role discovery, these three groups of nodes play the roles of *periphery*, *star* and *bridge*, respectively.

Fig. 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, skip-gram window size: 5. For node2vec, we set \(p = 1\) and \(q = 2\). For *struc2gauss*, number of walks per node is 20 and number of positive/negative nodes per node is 5. It can be observed that DeepWalk, LINE and GraRep fail to capture the global structural information. However, 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. 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. *struc2vec* can solve this problem to some extent. However, there is overlap between node 6 and 9. Our proposed *struc2gauss* outperforms all other methods. Both diagonal and spherical covariances can separate nodes based on global structural information and *struc2gauss* with spherical covariances performs better than diagonal covariances since it can recognize *star* and *bridge* nodes better.
5.2 Node 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 node clustering task to illustrate the potential of node representations learned by \textit{struc2gauss}. We use the latent representations learned by different methods (in \textit{struc2gauss}, we use means of learned Gaussian distribution) as features and K-means as the clustering algorithm to cluster nodes.

\textbf{Datasets.} We use two types of network data sets: networks with and without ground-truth clustering labels. For data with labels, to compare state-of-
Table 1: A brief introduction to data sets.

| Type     | Dataset   | # nodes | # edges | # groups |
|----------|-----------|---------|---------|----------|
| with labels | Brazilian-air | 131     | 1038    | 4        |
|          | European-air | 399     | 5993    | 4        |
| without labels | Arxiv GR-QC | 5242    | 28980   | 8        |
|          | Advogato   | 6551    | 51332   | 11       |
|          | Hamsterster| 2426    | 16630   | 10       |

Table 2: NMI for node clustering in air-traffic networks using different NE methods. In struc2gauss, EL and KL mean expected likelihood and KL divergence, respectively. D and S mean diagonal and spherical covariances, respectively. The highest value is in bold.

| Method          | Brazil | Europe | USA  |
|-----------------|--------|--------|------|
| DeepWalk        | 0.1303 | 0.0458 | 0.0766 |
| LINE            | 0.0684 | 0.0410 | 0.1088 |
| node2vec        | 0.0727 | 0.1722 | 0.0945 |
| GraRep          | 0.2097 | 0.1986 | 0.1811 |
| struc2vec       | 0.3758 | 0.2729 | 0.2486 |
| struc2gauss-EL-D| 0.5615 | 0.3234 | 0.3188 |
| struc2gauss-EL-S| 0.3796 | 0.2774 | 0.2967 |
| struc2gauss-KL-D| 0.5527 | 0.3145 | 0.3212 |
| struc2gauss-KL-S| 0.5675 | 0.3280 | 0.3217 |

Fig. 4: Goodness-of-fit of struc2vec and struc2gauss with different strategies and covariances on three real-world networks. Lower value means better performance.

For the data, we use air-traffic networks from [30] where the networks are undirected, nodes are airports, edges indicate the existence of commercial flights and labels correspond to their levels of activities. For data without labels, we select several real-world networks in different domains from Network Reposi-


A brief introduction to these data sets is shown in Table 1. Note that the numbers of groups for networks without labels are determined by MDL [13].

**Baselines.** We select several state-of-the-art NE algorithms as baselines, i.e., DeepWalk, LINE, GraRep, node2vec, and struc2vec. For our proposed struc2gauss, we test both diagonal and spherical covariances. In 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 LINE, the order of the proximity is 2. For node2vec, we set \( p = 1 \) and \( q = 2 \). For GraRep, the maximum matrix transition step is 3 and number of positive/negative nodes per node is 120.

**Evaluation Metric.** 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 \( C \) and ground-truth cluster \( D \):

\[
\text{NMI}(C, D) = \frac{2 * I(C, D)}{H(C) + H(D)},
\]

where the mutual information \( I(C, D) \) is defined as \( I(C, D) = H(C) - H(C|D) \) and \( H(\cdot) \) is the entropy.

For unlabeled networks, we use normalized *goodness-of-fit* as the evaluation metric. In *goodness-of-fit indices*, 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 SNs, nodes belonging to the same role are only approximately structurally equivalent. The essence of *goodness-of-fit indices* is to measure how approximate are the approximate structural equivalences. If the optimal model holds, then all nodes belonging to the same role are exactly structurally equivalent. *Goodness-of-fit* can measure how well the representation of roles and the relations among these roles fit a given network so this measure has been widely used in role discovery [27, 38]. 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 [38].

The NMI values for node clustering on networks with labels are shown in Table 2 and the normalized *goodness-of-fit* values for networks without labels are shown in Fig. 4. 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 [36].

- For baselines, *struc2vec* can capture the structural information to some extent since its performance is much better than DeepWalk and *node2vec*

\[\text{http://networkrepository.com/index.php}\]
Table 3: NMI for node clustering in air-traffic networks of Brazil, Europe and USA using struc2gauss with different similarity measures.

|         | Brazil-airport | Europe-airport | USA-airport |
|---------|----------------|----------------|-------------|
| SimRank | 0.1695         | 0.0524         | 0.0887      |
| MatchSim| 0.3574         | 0.2389         | 0.0913      |
| RoleSim | 0.5675         | 0.3280         | 0.3217      |

Fig. 5: Goodness-of-fit of struc2gauss with different similarity measures. Lower values are better.

while both of them fail in capturing the global structural information for node clustering.

Note that among the four different combinations of strategies, struc2gauss using KL divergence with spherical covariance performs best on all networks. In following sections, we only test the combination of KL divergence and spherical covariance in struc2gauss if not explicitly stated otherwise.

5.3 Influence of Similarity Measures

To analyze the influence of different similarity measures on learning node representations, we compare two different measures for global structural similarity, i.e., SimRank [15] and MatchSim [21], to RoleSim which is by default used in our framework. The data sets and evaluation metrics used in this experiment are the same to Section 5.2.

The NMI values for networks with labels are shown in Table 3 and the goodness-of-fit values are shown in Fig. 5. 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 2). 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 [22].

5.4 Uncertainty Modeling

We use stochastic blockmodels [17] to generate synthetic networks. In specific, 
we generate a network with 200 nodes and 4 blocks and the original network 
can be clustered into these 4 blocks perfectly. Then we add different numbers 
of edges to the networks step by step randomly from 100 to 1000 with the 
interval 100. Totally we have one original network and 100 evolved networks 
with different levels of noise. We learn node representations and corresponding 
uncertainties reflected by covariances using struc2gauss. Since we select the 
spherical and diagonal covariance in our experiments, we compute the traces 
of covariance matrices to compare the uncertainties in different embeddings.

The comparison is shown in Fig. 6 where it can be observed that with 
more noise being added to the network, larger traces of covariance matrices 
we have. When there is less noise (less than 400 edges have been added), the 
differences between original network and evolved networks are not obvious, but 
with more noise introduced (more than 400 edges have been added), the dif-
f erences become significant. This demonstrates that our proposed struc2gauss 
can capture the uncertainties of learned node representations.

5.5 Parameter Sensitivity

We consider three major types of parameters in struc2gauss, i.e., latent di-
mensions, number of samples per node and number of positive/negative nodes 
per node. In order to evaluate how changes to these parameters affect per-
formance, we conducted the same node clustering experiment on the labeled 
USA air-traffic network introduced in Section 5.2.

In the interest of brevity, we first fix two parameters and then vary the 
third one. 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. The results of parameter sensitivity are shown in 
Fig. 7. It can be observed from Fig. 7(a) and 7(b) that the trends are relatively 
stable, i.e., the performance is insensitive to the changes of representation di-
nensions and numbers of samples. The performance of clustering is improved 
with the increase of numbers of positive/negative nodes shown in Fig. 7(c). 
Therefore, we can conclude that struc2gauss is more stable than other meth-
ods. It has been reported that other methods, e.g., DeepWalk [29], LINE [34]
and node2vec [10], 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.

Note that we observed the same trend in other networks so only results on USA-airport network are shown here.
6 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 graph clustering task, i.e., role discovery, on both synthetic and 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.

We conclude by indicating promising directions for further study. A first area for improvement is to study faster and more scalable structural similarity calculation method. Since we care more about the most similar nodes given a query node, an approximate method to calculate the structural similarity may also be a promising direction. Second, it is interesting to extend our method to different scenarios, e.g., dynamic networks and heterogeneous networks. Many real-world networks are dynamic with evolving structures and heterogeneous with different types of nodes and edges. How to learn representations in such scenarios based on struc2gauss will be a challenging and meaningful problem. A third area for future exploration is to exploit other methods which can calculate structural similarity or capture structural information.

References

1. Airoldi, E.M., Blei, D.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels. Journal of Machine Learning Research 9(Sep), 1981–2014 (2008)
2. Antonellis, I., Molina, H.G., Chang, C.C.: Simrank++: query rewriting through link analysis of the click graph. Proceedings of the VLDB Endowment 1(1), 408–421 (2008)
3. Bojchevski, A., Günnemann, S.: Deep gaussian embedding of attributed graphs: Unsupervised inductive learning via ranking. arXiv preprint arXiv:1707.03815 (2017)
4. Borgatti, S.P., Everett, M.G.: Two algorithms for computing regular equivalence. Social networks 15(4), 361–376 (1993)
5. Cao, S., Lu, W., Xu, Q.: Grarep: Learning graph representations with global structural information. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 891–900. ACM (2015)
6. Dos Santos, L., Piwowarski, B., Gallinari, P.: Multilabel classification on heterogeneous graphs with gaussian embeddings. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 606–622. Springer (2016)
7. Fortunato, S.: Community detection in graphs. Physics reports 486(3), 75–174 (2010)
8. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855–864. ACM (2016)
9. Hamilton, W.L., Ying, R., Leskovec, J.: Representation learning on graphs: Methods and applications. IEEE Data Eng. Bull. 40(3), 52–74 (2017)
10. He, S., Liu, K., Ji, G., Zhao, J.: Learning to represent knowledge graphs with gaussian embedding. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 623–632. ACM (2015)
11. Henderson, K., Gallagher, B., Eliassi-Rad, T., Tong, H., Basu, S., Akoglu, L., Koutra, D., Faloutsos, C., Li, L.: Rolex: structural role extraction & mining in large graphs. In: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1231–1239. ACM (2012)
12. Jin, R., Lee, V.E., Hong, H.: Axiomatic ranking of network role similarity. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 922–930. ACM (2011)
13. Karrer, B., Newman, M.E.: Stochastic blockmodels and community structure in networks. Physical Review E 83(1), 016107 (2011)
14. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. Journal of the ACM (JACM) 46(5), 604–632 (1999)
15. Li, J., Dani, H., Hu, X., Tang, J., Chang, Y., Liu, H.: Attributed network embedding for learning in a dynamic environment. arXiv preprint arXiv:1706.01860 (2017)
16. Liben-Nowell, D., Kleinberg, J.: The link-prediction problem for social networks, journal of the Association for Information Science and Technology 58(7), 1019–1031 (2007)
17. Lin, Z., Lyu, M.R., King, I.: Matchsim: a novel neighbor-based similarity measure with maximum neighborhood matching. In: Proceedings of the 18th ACM conference on Information and knowledge management, pp. 1613–1616. ACM (2009)
18. Lyu, T., Zhang, Y., Zhang, Y.: Enhancing the network embedding quality with structural similarity. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 147–156. ACM (2017)
19. Ma, Y., Wang, S., Ren, Z., Yin, D., Tang, J.: Preserving local and global information for network embedding. arXiv preprint arXiv:1710.07266 (2017)
20. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
21. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp. 3111–3119 (2013)
22. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Tech. rep., Stanford InfoLab (1999)
27. Pei, Y., Zhang, J., Fletcher, G., Pechenizkiy, M.: Dynmf: Role analytics in dynamic social networks. Twenty-Seventh International Joint Conference on Artificial Intelligence (2018)
28. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532–1543 (2014)
29. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 701–710. ACM (2014)
30. Ribeiro, L.F., Saverese, P.H., Figueiredo, D.R.: struc2vec: Learning node representations from structural identity. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 385–394. ACM (2017)
31. Rossi, R.A., Ahmed, N.K.: Role discovery in networks. IEEE Transactions on Knowledge and Data Engineering 27(4), 1112–1131 (2015)
32. Shaw, B., Jebara, T.: Structure preserving embedding. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 937–944. ACM (2009)
33. Tang, J., Qu, M., Mei, Q.: 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, pp. 1165–1174. ACM (2015)
34. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1067–1077. International World Wide Web Conferences Steering Committee (2015)
35. Tenenbaum, J.B., De Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. science 290(5500), 2319–2323 (2000)
36. Vilnis, L., McCallum, A.: Word representations via gaussian embedding. arXiv preprint arXiv:1412.6623 (2014)
37. Wang, S., Aggarwal, C., Tang, J., Liu, H.: Attributed signed network embedding. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 137–146. ACM (2017)
38. Wasserman, S., Faust, K.: Social network analysis: Methods and applications, vol. 8. Cambridge university press (1994)
39. Zhao, P., Han, J., Sun, Y.: P-rank: a comprehensive structural similarity measure over information networks. In: Proceedings of the 18th ACM conference on Information and knowledge management, pp. 553–562. ACM (2009)