Structural Deep Network Embedding

Abstract

1. **Background**:Existing network embedding methods adopt shallow models. **Problem**:Since underlying network structure is complex, shallow models cannot capture the highly non-linear network structure, resulting in sub-optimal network representation. Find a method that is able to effectively capture the highly non-linear network structure and preserve simultaneously the global and local structure.
2. **Method**: *SDNE*,a semi-supervised deep model .which has multiple layers of non-linear functions. Exploit the *first-order* and *second-order* proximity jointly to preserve the network structure. The *second-order* proximity is used by unsupervised component to capture the global network structure. The *first-order* proximity is used as supervised information in the supervised component to preserve the local network structure.
3. Three application ,i.e. multi-label classification, link prediction and visualization.

Introduction

1. Learning network representations faces the following great challenges. (1) **High non-linearity** (2) **Structure-preserving**. (3) **Sparsity** . In order to address these problems, propose to exploit the first-order and second-order proximity jointly into the learning process. **The first-order proximity is the local pairwise similarity only between the vertexes linked by edges**. **The second-order proximity indicates the similarity of the vertexes' neighborhood structures, to capture the global network structure.**
2. The method, as the abstract shows.

Structural Deep Network Embedding

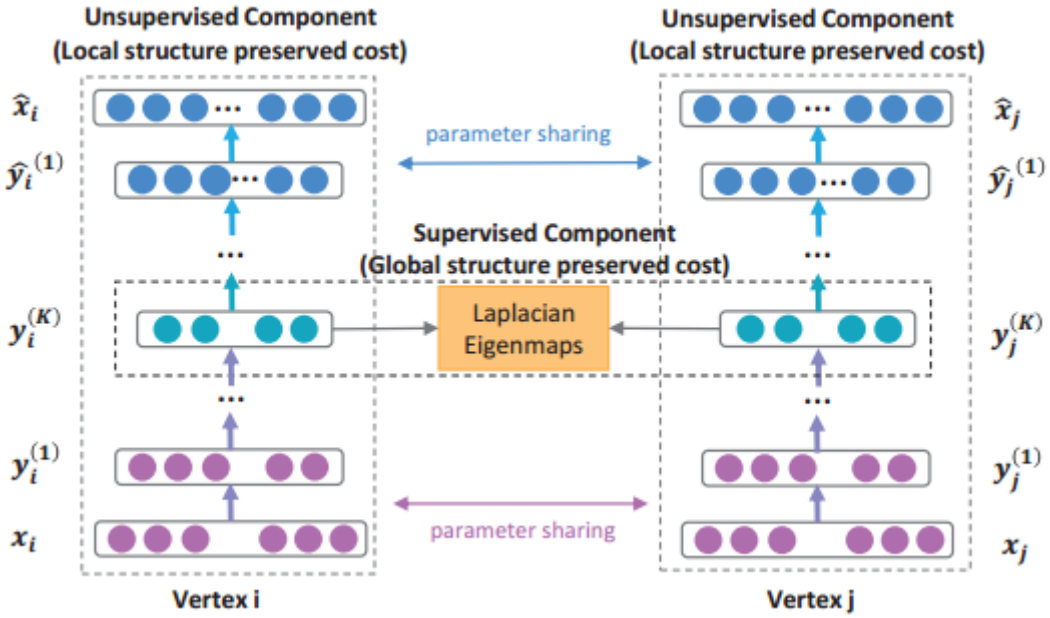
Definition

1. *first-order-proximity*. For any pair of vertexes, if $s_{i,j}$>0, there exists positive first-order proximity between v_i and v_j .
2. *second-order-proximity*. Describes the proximity of the pair's neighborhood structure. Let $\mathcal{N}_u = \{s_{u,1}, \ldots, s_{u,|V|}\}$ denote the *first-order-proximity* between v_u and other vertexes. Then, *second-order-proximity* is determined by the similarity of $\mathcal{N}_u$ and $\mathcal{N}_v$. Intuitively, the *second-order proximity* assumes that if two vertexes share many common neighbors, they tend to be similar.

The Model

Framework

As the method of abstract shows.



Loss Function

1. The *second-order* proximity refers to how similar the neighborhood structure of a pair of vertexes is. It is an unsupervised model which is composed of two parts, i.e. the encoder and decoder. **The encoder** consists of multiple non-linear functions that map the input data to representation space. **The decoder** also consists of multiple non-linear function mapping the representations in representation space to reconstruction space. Then given the input $\mathbf{x}_i$, the hidden representations for each layer are shown as follows:

$$\mathbf{y}_i^{(1)} = \sigma\left(W^{(1)}\mathbf{x}_i + \mathbf{b}^{(1)}\right)$$
$$\mathbf{y}_i^{(k)} = \sigma\left(W^{(k)}\mathbf{y}_i^{k-1} + \mathbf{b}^{(j)}\right), k = 2, \ldots, K \qquad (1)$$

After obtaining $\mathbf{y}_i^K$, we can obtain the output $\hat{\mathbf{x}}_i$ by reversing the calculation process of encoder. The goal of the autoencoder is to **minimize the reconstruction error of the output and the input.** The loss function is shown as follows:

$$\mathcal{L} = \sum_{i=1}^{n} ||\hat{\mathbf{x}}_i - \mathbf{x}_i||_2^2 \qquad (2)$$

In the network, we can observe some links but simultaneously many legitimate links are not observed, which means that the links between vertexes do indicate their similarity but no links do not necessarily indicate their dissimilarity. Moreover, duo to the sparsity of networks, the number of non-zero elements in S is far less than that of zero elements. To address this problem, we impose penalty to the reconstruction error of the non-zero elements than that of zero elements. The revised objective function is shown as follows:

$$\begin{aligned}\mathcal{L}_{2nd} &= \sum_{i=1}^{n} \|(\hat{\mathbf{x}}_i - \mathbf{x}_i) \odot \mathbf{b}_i\|_2^2 \\ &= \|(\hat{X} - X) \odot B\|_F^2\end{aligned} \qquad (3)$$

If $s_{i,j} = 0, b_{i,j} = 1, else, b_{i,j} = \beta > 1.$

2. The *second-order* proximity can be regarded as the supervised information to constrain the similarity of the latent representation of a pair of vertexes. The loss function for this goal is defined as follow:

$$\begin{aligned}\mathcal{L}_{1st} &= \sum_{i,j=1}^{n} s_{i,j} \|\mathbf{y}_i^{(K)} - \mathbf{y}_j^{(K)}\|_2^2 \\ &= \sum_{i,j=1}^{n} s_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|_2^2\end{aligned} \qquad (4)$$

The Eq. (4) borrows the idea of Laplacian Eigenmaps, which incurs a penalty when similar vertexes are mapped far away in the embedding space.

3. Combines Eq. (3) and Eq. (4) and joint minimizes the following objective function:

$$\mathcal{L}_{mix} = \mathcal{L}_{2nd} + \alpha\mathcal{L}_{1st} + v\mathcal{L}_{reg} = \|(\hat{X} - X) \odot B\|_F^2 + \alpha \sum_{i,j=1}^{n} s_{i,j}\|\mathbf{y}_i - \mathbf{y}_j\|_2^2 + v\mathcal{L}_{reg} \quad (5)$$

$$\mathcal{L}_{reg} = \frac{1}{2}\sum_{k=1}^{K}(\|W^{(k)}\|_F^2 + \|\hat{W}^{(k)}\|_F^2)$$

Analysis and Discussion

New vertexes

Training Complexity