Network Embedding

Wenbao Li

Data Mining Lab

April 9, 2017

Content

1. Preface
2. Introduction
3. Network Embedding Models
   - Word2Vec(Intro)
   - DeepWalk
   - HNE
   - LINE
   - Node2Vec
   - SDNE(Intro.)
   - LANE(Intro)
4. Conclusion
5. Material
6. References

Preface

为什么讲这个TOPIC

- 之前做异构网络聚类的时候，感觉就是在找一个vector representation，虽然比较low的选择了最简单的方法。后来看到几篇CMF、或者Multi-View NMF等相关的文章，本质上也是在找这样一个vector representation，不管它之前是node，还是word等，或者就是一个vector。
- 其次是看到近几年KDD上这方面的文章很多。

过程中的问题

上面说了，这儿的坑挺深的。

- 其一，最基础的模型好像都是来源于NLP中的word embedding。所以背景不强大，看起来很吃力。
- 其二，借鉴SkipGram模型的方法都是深度学习相关的。
- 其三，关于好几个相关topic的对比，它们的关系（后面会讨论——真的是讨论，答案我也不知道）。

该方向有什么值得探索的地方?

- 异构网络embedding。
- 利用网络结构进行多源数据融合（比如下面HNE中就是），学习一个representation。
- 结合半监督方法来学习一个对各类任务都有效的representation。

Introduction

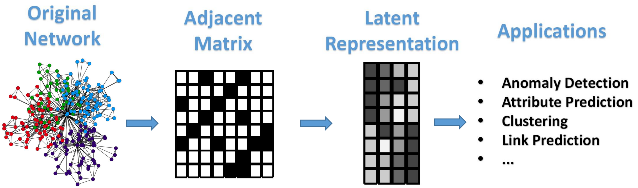
What is network/ graph embedding?

Also network representation

Definition (Network/Graph embedding)

Map each node in a graph/network into a low-dimensional space.

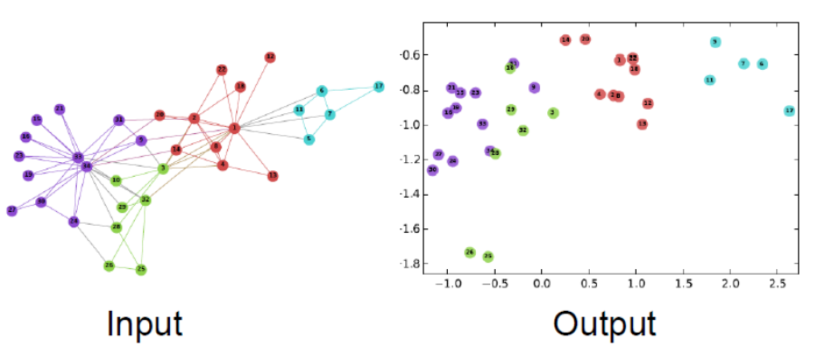
- Distributed representation for nodes
- Similarity between nodes indicates the link strength
- Encode network information and generate node representation



What is network/ graph embedding?

Example

Example (Zachary's Karate Network)

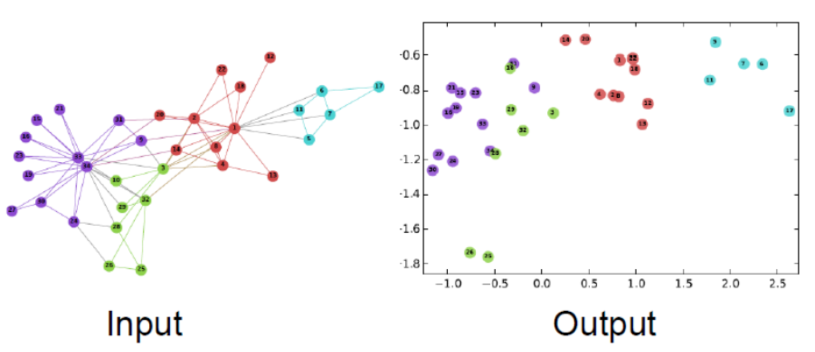


Do you know some familiar network embedding techniques?

What is network/ graph embedding?

Example

Example (Zachary's Karate Network)



Do you know some familiar network embedding techniques?

- Laplacian Embedding → Spectral Clustering

Laplacian Embedding
Intro.

- Embed the graph in a k-dimensional Euclidean space. The embedding is given by the $n \times k$ matrix $\mathbf{F} = [\mathbf{f}_1 \mathbf{f}_2 \dots \mathbf{f}_k]$ where the i-th row of this matrix $\mathbf{f}^{(i)}$ corresponds to the Euclidean coordinates of the i-th graph node v_i.
- We need to minimize (Belkin & Niyogi '03):

$$\text{argmin}_{\mathbf{f}_1 \mathbf{f}_2 \dots \mathbf{f}_k} \sum_{i,j=1}^{n} w_{ij} |\mathbf{f}^{(i)} - \mathbf{f}^{(j)}|^2 \text{with: } \mathbf{F}^T \mathbf{F} = \mathbf{I}. \tag{1}$$

- The solution is provided by the matrix of eigenvectors corresponding to the k lowest nonzero eigenvalues of the eigenvalue problem $\mathbf{Lf} = \lambda \mathbf{f}$ where $L = D - A$.

Discuss: Is there any **differences** from these topics?

- Representation/Feature learning
- Dimensionality reduction
- Subspace learning
- ...

Have a look at these definitions

Representation/Feature learning

Definition (Representation/Feature learning (from wiki))

In machine learning, feature learning or representation learning[1] is a set of techniques that learn a feature: a transformation of raw data input to a representation that can be effectively exploited in machine learning tasks.

- Superv.: supervised neural networks, multilayer perceptron, and (supervised) dictionary learning
- Unsuperv.: dictionary learning, **ICA**, **PCA**, LLE, **autoencoders**, matrix factorization, and various forms of clustering.

Maybe Feature Learning $\supseteq$ Network Embedding.

Have a look at these definitions

Dimensionality reduction

Definition (Dimensionality reduction (from wiki))

In machine learning and statistics, dimensionality reduction or dimension reduction is the process of reducing the number of random variables under consideration,[1] via obtaining a set of principal variables. It can be divided into feature selection and **feature extraction**

E.g.: **ICA**, Isomap, Kernel PCA, Latent semantic analysis, Partial least squares, **PCA**, Multifactor dimensionality reduction, Nonlinear dimensionality reduction, Multilinear PCA, Multilinear **subspace learning**, Semidefinite embedding, **autoencoder**, Deep feature synthesis.

Most dimensionality reduction methods can be applied in network embedding.(I think)

Have a look at these definitions

Subspace learning

Definition (Subspace learning)

to transform the original input features to a lower dimensional subspace.

There are: PCA, LDA, Locality Preserving Projection (LPP), Neighborhood Preserving Embedding (NPE) and so on.

Why learn embed?

Let's go back to word embedding

Definition (Word Embedding)

Word embedding is the collective name for a set of language modeling and **feature learning** techniques in natural language processing (NLP) where words or phrases from the vocabulary are mapped to vectors of real numbers. Conceptually it involves a mathematical embedding from a space with one dimension per word to a **continuous vector space** with much **lower** dimension.

Methods to generate this mapping include neural networks, **dimensionality reduction** on the word co-occurrence matrix, probabilistic models,and explicit representation in terms of the context in which words appear.
Word and phrase embeddings, when used as the underlying input representation, have been shown to **boost the performance** in NLP tasks such as syntactic parsing and sentiment analysis.

Why learn embedding? The Motivation

Network ∼ Text

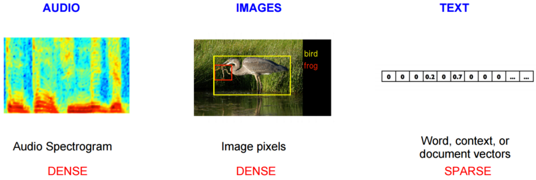


Image and audio processing systems work with rich, high-dimensional datasets encoded as vectors of the individual raw pixel-intensities for image data.
However, natural language processing systems traditionally treat words as discrete atomic symbols
e.g. 'cat': Id537, 'dog': Id143 ⇒ **arbitrary**,**no useful information**, **sparsity**
Using vector representations can overcome some of these obstacles.
Most networks are sparse too.

Intro: How to embed?

- Goal 1: Reconstruct the original network
- Goal 2: Support network inference
  - Reflect network structure
  - Maintain network properties

Network Embedding Models

Word2Vec

- **Input:** a sequence of words from a vocabulary V.
- **Output:** a fixed-length vector for each term in the vocabulary v_w

(Dataset)

Given: the quick brown fox jumped over the lazy dog
Formed as $<$ **context, target** $>$ *pairs.*
([the, brown], quick), ([quick, fox], brown), ([brown, jumped], fox), ([for, over], jumped), ([jumped, the], over), ...

Word2vec

Architecture 1: CBOW

- **CBOW:** predicts the current word using surrounding contexts
  - $Pr(w_t|\text{context}(w_t))$
    - ★ Window size 2c
    - ★ $\text{context}(w_t) = [w_{t-c}, \ldots, w_{t+c}]$
  - Using a K-dimensional vector to represent words
    - ★ $w_t \to v_{w_t}$
    - ★ $\bar{v}_{w_t} = \frac{\sum\limits_{i=t-c}^{t+c} v_{w_i}}{2c}, i \neq t$
  - Finally, use softmax function to classify.

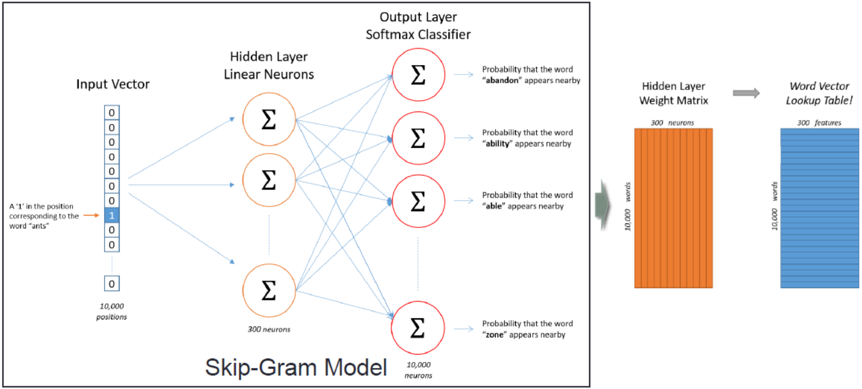
$$Pr(w|w_{context}) = \frac{\exp\left(sim(\bar{v}_w, v_w)\right)}{\sum_{w'} \exp\left(sim(\bar{v}_w, v_{w'})\right)} \tag{2}$$

- **SkipGram:** predicts surrounding contexts using the current word
  - $Pr(\text{context}(w_t)|w_t)$
    - Window size 2c
    - context$(w_t) = [w_{t-c}, \ldots, w_{t+c}]$
  - Finally, use softmax function to classify.

$$Pr(w'|w) = \frac{\exp\left(sim(v_w, v_{w'})\right)}{\sum_{w''} \exp\left(sim(v_w, v_{w''})\right)} \tag{3}$$

E.g. :(quick, the), (quick, brown), (brown, quick), (brown, fox)...

SkipGram

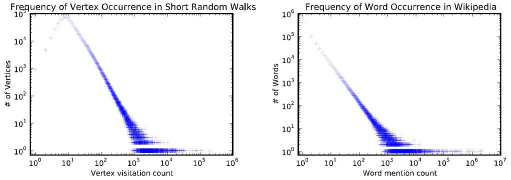


From Chris McCormick

DeepWalk
(Perozzi. *et al* KDD 2014[1])

Motivation

The similarity(Vertex VS. Word)



(a) YouTube Social Graph (b) Wikipedia Article Text

Figure 2: The power-law distribution of vertices appearing in short random walks (2a) follows a power-law, much like the distribution of words in natural language (2b).

So, a transfer from language model to network model.

DeepWalk

(Perozzi. *et al* KDD 2014[1])

Basic Idea

Transfer:

- *words* $\leftrightarrow$ *vertices*
- *sentences* $\leftrightarrow$ *vertex sequences*
- *articles* $\leftrightarrow$ *network*

Input:

- *corpus: a set of short truncated random walks.*
- *vocabulary: vertices*

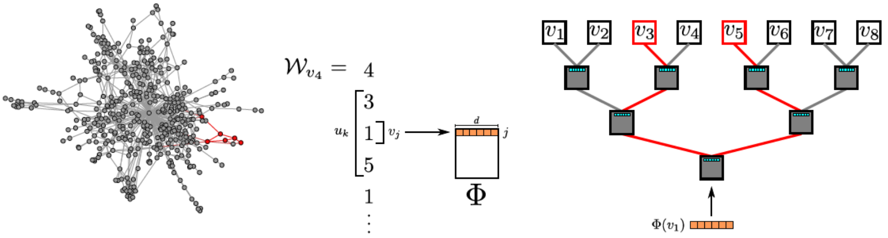
Output:

- *Vertex representation*

DeepWalk

(Perozzi. *et al* KDD 2014[1])

Framework



(a) Random walk generation. (b) Representation mapping. (c) Hierarchical Softmax.

Figure 3: Overview of DeepWalk. We slide a window of length $2w+1$ over the random walk $\mathcal{W}_{v_4}$, mapping the central vertex v_1 to its representation $\Phi(v_1)$. Hierarchical Softmax factors out $\Pr(v_3 \mid \Phi(v_1))$ and $\Pr(v_5 \mid \Phi(v_1))$ over sequences of probability distributions corresponding to the paths starting at the root and ending at v_3 and v_5. The representation Φ is updated to maximize the probability of v_1 co-occurring with its context $\{v_3, v_5\}$.

DeepWalk

(Perozzi. *et al* KDD 2014[1])

Algorithm 1 DeepWalk(G, w, d, γ, t)

Input: graph $G(V, E)$
window size w
embedding size d
walks per vertex γ
walk length t
Output: matrix of vertex representations $\Phi \in \mathbb{R}^{|V| \times d}$

Initialization: Sample Φ from $\mathcal{U}^{|V| \times d}$
Build a binary Tree T from V
for $i = 0$ to γ **do**
  $\mathcal{O} = \text{Shuffle}(V)$
  for each $v_i \in \mathcal{O}$ **do**
    $\mathcal{W}_{v_i} = RandomWalk(G, v_i, \text{t})$
    SkipGram$(\Phi, \mathcal{W}_{v_i}, w)$
  end for
end for

DeepWalk
(Perozzi. *et al* KDD 2014[1])

Algorithm 2 SkipGram(Φ, $\mathcal{W}_{v_i}$, w)

1: **for each** $v_j \in \mathcal{W}_{v_i}$ **do**
2: **for each** $u_k \in \mathcal{W}_{v_i}[j - w : j + w]$ **do**
3: $J(\Phi) = -\log \Pr(u_k \mid \Phi(v_j))$
4: $\Phi = \Phi - \alpha * \frac{\partial J}{\partial \Phi}$
5: **end for**
6: **end for**

DeepWalk

(Perozzi. *et al* KDD 2014[1])

How to calculate $Pr(u_k|\Phi(v_j))$ efficiently?

In the model: Hierarchical Softmax.

$$Pr(u_k|\Phi(v_j)) = \prod_{i=1}^{\lceil log|V| \rceil} Pr(b_l|\Phi(v_j)) \tag{4}$$

$Pr(b_l|\Phi(v_j))$ could be modeled by a binary classifier that is assigned to the parent of the node b_l.

实际上，我对这个有点疑问：怎么计算$Pr(b_l|\Phi(v_j))$?

计算方法[?]:

$$Pr(w|w_I) = \prod_{j=1}^{L(w)-1} \sigma\left(\langle n(w,j+1) = ch(n(w,j))\rangle \cdot {v'}^{T}_{n(w,j)} v_{w_I}\right) \tag{5}$$

where $\langle x \rangle$ be 1 if x is true and -1 otherwise.

DeepWalk
(Perozzi. *et al* KDD 2014[1])

Analysis

1. 优点：将语言模型应用到网络中。
2. *(Ques)*说好的深度学习是指什么？？（后来想明白了，就是指*SkipGram*模型）
3. *(Ques)*随机游走生成器：它的生成规则→某些结点产生的*walks*势必会重新经过起点。而文中也说明这样的*walks*是不太好的。
4. *(Ques)*对于比较离散的结点（噪声点），显然各种*walk*都到达不了它，它自己也*walk*不出去，那你怎么学习它的*representation*? 仅仅在简单地初始化之后就不管它？是否合理？应该怎么处理？
5. *(Ques)*对于很多相似性学习模型，大多数关注的仅仅只有相似性信息，而不去维护原始的不相似信息，是否合理？

HNE

Chang. *et al* KDD 2015[2]

Heterogeneous Network Embedding via Deep Architectures

Basic Idea

1. *Deep Learning: Feature Representation and Nonlinear embedding*
   - *Image: CNN(Convolution Neural Network)*
   - *Text: TF-IDF pre-process + FC(fully connected)*
2. *Matrix Transformation: Linear embedding*

HNE

CHANG. *et al* KDD 2015[2]

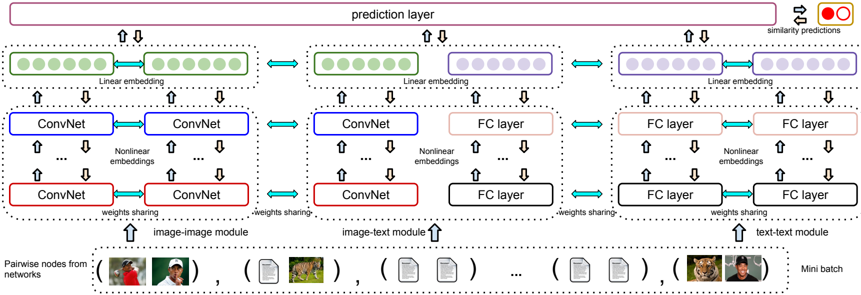


FIGURE: An illustration of HNE

HNE

CHANG. *et al* KDD 2015[2]

BASIC IDEA (HOW TO EMBED)

Transform vector representation:

$$s(x_i, x_j) = \bar{x}_i^T \bar{x}_j = (U^T x_i)^T U^T x_j = x_i^T M_{II} x_j \tag{6}$$

$$s(z_i, z_j) = \bar{z}_i^T \bar{z}_j = (V^T z_i)^T V^T z_j = z_i^T M_{TT} z_j \tag{7}$$

$$s(x_i, z_j) = \bar{x}_i^T \bar{z}_j = (U^T x_i)^T V^T z_j = x_i^T M_{IT} z_j \tag{8}$$

Then define a distance function:

$$d(x_i, x_j) = s(x_i, x_j) - t_{II} \tag{9}$$

Define loss function which can be seen as a binary logistic regression guided by network linkages.

$$L(x_i, x_j) = log(1 + \exp(-A_{i,j} d(x_i, x_j))) \tag{10}$$

HNE

Chang. *et al* KDD 2015[2]

The linear embedding optimization objective:

Optimization Objective

$$min_{U,V} \frac{1}{N_{II}} \sum_{v_i, v_j \in \mathcal{V}_I} L(x_i, x_j) + \frac{\lambda_1}{N_{TT}} \sum_{v_i, v_j \in \mathcal{V}_T} L(z_i, z_j) + \frac{\lambda_2}{N_{IT}} \sum_{v_i \in \mathcal{V}_I, v_j \in \mathcal{V}_T} L(x_i, z_j) + \lambda_3 (\|U\|_F^2 + \|V\|_F^2) \tag{11}$$

HNE

CHANG. *et al* KDD 2015[2]

The optimization objective:

OPTIMIZATION OBJECTIVE (DEEP LEARNING VERSION)

$$
\begin{aligned}
min_{U,V,D_I,D_T} \frac{1}{N_{II}} \sum_{v_i,v_j \in \mathcal{V}_I} & L(p_{D_I}(X_i), p_{D_I}(X_j)) + \lambda_3(\|U\|_F^2 + \|V\|_F^2) \\
+ \frac{\lambda_1}{N_{TT}} \sum_{v_i,v_j \in \mathcal{V}_T} & L(q_{D_T}(z_i), q_{D_T}(z_j)) \\
+ \frac{\lambda_2}{N_{IT}} \sum_{v_i \in \mathcal{V}_I, v_j \in \mathcal{V}_T} & L(p_{D_I}(X_i), q_{D_T}(z_j))
\end{aligned} \tag{12}
$$

HNE

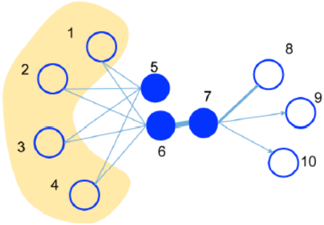
CHANG. *et al* KDD 2015[2]

ANALYSIS

1. 算法的新颖点在于：将深度学习引入异构网络这个框架里。同时学习*representation*与*embedding*，且互相增强。重点在于深度学习能够学习非线性的*embedding*
2. *Target: images and texts which contain rich information themselves.* 对于一般的网络（只有节点与连接信息），需要有事先有一个向量表征，如何获取？或者说这个方法只适应于图像文本这类适合用深度学习来处理的对象吗？

LINE

TANG. *et al* WWW 2015[3]



- First-order Proximity: pairwise proximity, direct links. E.g.: $< 6, 7 >$
- Second-order Proximity: pairwise proximity, similarity between neighbors. E.g. $< 5, 6 >$

Challenges:

- Deal with Large-scale network
- Preserve the first and second-order proximity in various types of networks (undirected, directed, weighted)

LINE

Tang. *et al* WWW 2015[3]

(First-order Proximity)

First, consider a predictive probability

$$p_1(v_i, v_j) = \frac{1}{1 + \exp(-\vec{u}_i^T \cdot \vec{u}_j)} \tag{13}$$

where $p(\cdot,\cdot)$ *is a joint distribution.*
And $\hat{p}_1(i,j) = \frac{w_{ij}}{W}, W = \sum\limits_{(i,j)\in E} w_{ij}$ *is the empirical probability.*
Distance of distribution:

$$O_1 = d(\hat{p}_1(\cdot,\cdot), p_1(\cdot,\cdot)) \tag{14}$$

Here, use KL-divergence.

$$O_1 = -\sum_{(i,j)\in E} w_{ij} \log p_1(v_i, v_j) \tag{15}$$

LINE

TANG. *et al* WWW 2015[3]

(SECOND-ORDER PROXIMITY)

First, given two denotations $\hat{u}_i$ *and* $\hat{u}'_i$.
Define predictive probability of "context" v_j *generated by vertex* v_i*:*

$$p_2(v_j|v_i) = \frac{\exp\left(\hat{u}_j'^T \cdot \hat{u}_i\right)}{\sum_{k=1}^{|V|} \exp\left(\hat{u}_k'^T \cdot \hat{u}_i\right)} \tag{16}$$

LINE

TANG. *et al* WWW 2015[3]

(Second-order Proximity)

Empirical probability distribution:

$$\hat{p}_2(\cdot|v_i) = \frac{w_{ij}}{d_i} \tag{17}$$

d_i *is the out-degree of vertex* i, $d_i = \sum_{k \in N(i)} w_{ik}$.
Distance of distribution:

$$O_2 = \sum_{i \in V} \lambda_i d(\hat{p}_2(\cdot|v_i), p_2(\cdot|v_i)) \tag{18}$$

Here, use KL-divergence, and set $\lambda_i = d_i$

$$O_2 = -\sum_{(i,j) \in E} w_{ij} \log p_2(v_j|v_i) \tag{19}$$

LINE

TANG. *et al* WWW 2015[3]

How to learn:

- train the LINE model which preserves the first-order proximity and second-order proximity separately and then concatenate the embeddings trained by the two methods for each vertex.
- jointly train the two objective functions.

Node2Vec

Grover. *et al* KDD 2016[4]

Motivation:

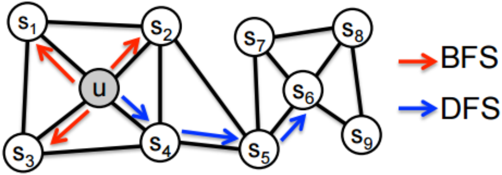


Figure 1: BFS and DFS search strategies from node u ($k = 3$).

- $< u, s_1 >$: Homogeneity
- $< u, s_6 >$: Structural equivalence

NODE2VEC
GROVER. *et al* KDD 2016[4]

Biased Random Walk:

$$Pr(c_i = x|c_{i-1} = v) = \frac{\pi_{v,x}}{Z}, if(v,x) \in E \tag{20}$$

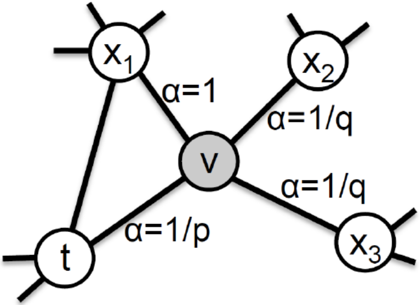
The transition probability $\pi_{v,x} = \alpha_{pq}(t,x) \cdot w_{vx}$
The search strategy:

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} & \text{if } d_{tx} = 0 \\ 1 & \text{if } d_{tx} = 1 \\ \frac{1}{q} & \text{if } d_{tx} = 2 \end{cases} \tag{21}$$

where p is **return parameter**, q is **in-out parameter**

Node2Vec

Grover. *et al* KDD 2016[4]



Biased Random Walk:

- Parameters p, q controls interpolation between DFS and BFS
- Semi-supervised learning: decide p, q from labeled nodes
- Similar optimization model as DeepWalk

Node2Vec

Grover. *et al* KDD 2016[4]

Analysis (Node2vec)

- 同时考虑同质性与结构等价性。
- 将*BFS*与*DFS*结合为一个新的有偏随机游走。

Node2Vec

Grover. *et al* KDD 2016[4]

Analysis (DeepWalk v.s. Node2vec)

- 类比于*word embedding*，分别对节点的上下文环境（*neighborhood*）进行了定义。
- 都是生成方法。
- 没有显式的优化目标。

SDNE(Intro.)

Wang. *et al* KDD 2016[5]

Idea:

- Highly non-linear: Deep learning method
- Structure-preserving and Sparse: jointly train first-order and second-order proximity in LINE.
- Semi-supervised architecture: 1st-order(local) with supervised component while 2nd-order(global) with unsupervised component

SDNE(Intro.)

Wang. *et al* KDD 2016[5]

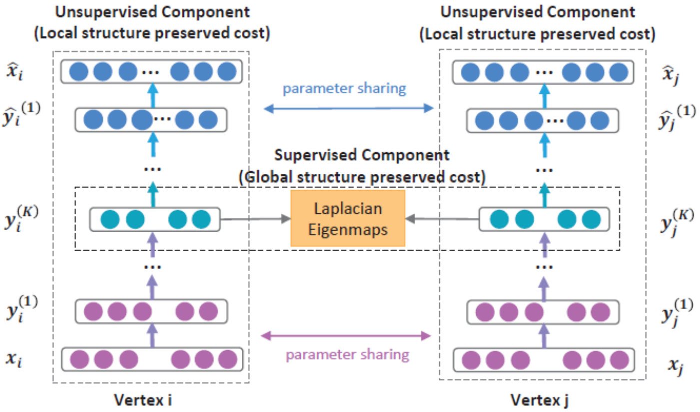


Figure 2: The framework of the semi-supervised deep model of *SDNE*

LANE(Intro.)

Huang. *et al* WSDM 2017[6]

- **Focus**: Attributed network(node-to-node links and extra features)
- **Target**: Leveraging both network proximity and node attribute affinity, at the same time, combining label information
- **Challenges**: sparse, incomplete, noisy, heterogeneous
- **Idea**: A supervised manner: with label information incorporated

LANE(Intro.)

Huang. *et al* WSDM 2017[6]

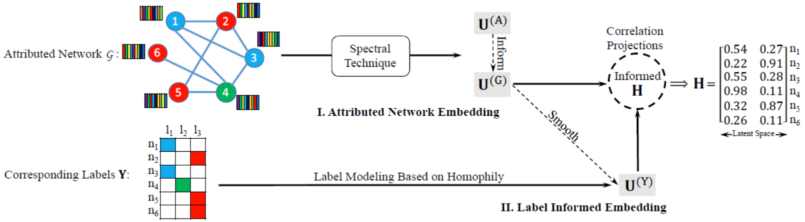


Figure 1: LANE maps the node proximities in attributed network into $U^{(G)}$ and label proximity into $U^{(Y)}$, as well as incorporates them into a joint embedding representation H via correlation projections.

Analysis

主要是有监督学习，实验展示了分类效果。不知道在其他无监督任务上的效果是否依然很好?

Conclusion

CONCLUSION

1. Network embedding with deep learning methods.
2. Network embedding by preserving network structures such as:
   - Node context
   - Pair-wise proximity
   - Community structures
3. Network embedding with (or without) label information.
4. Heterogeneous network embedding.

Some materials I

- 附件1：T3A 社会计算part3.pdf
- 附件2：Network Representation Tutorial.pdf by Peng Cui(http://media.cs.tsinghua.edu.cn/~multimedia/cuipeng/)
- Word2vec homepage(https://www.tensorflow.org/versions/r0.12/tutorials/word2vec/index.html)
- 附件3：GE4TE, 用Network embedding来辅助text embedding
- Related works
- A list of network embedding from github(https://github.com/chihming/awesome-network-embedding)
- DeepWalk: home page(http://www.perozzi.net/projects/deepwalk/)

Some materials II

- DeepWalk: source code(https://github.com/phanein/deepwalk)
- LINE: source code(https://github.com/tangjianpku/LINE)
- Node2vec: home page(http://snap.stanford.edu/node2vec/)
- Node2vec: source code(https://github.com/snap-stanford/snap/tree/master/examples/node2vec)
- SDNE知乎专栏阅读笔记(https://zhuanlan.zhihu.com/p/24769965)
- 知乎专栏阅读paper笔记(https://zhuanlan.zhihu.com/p/24328777)，好多方向，不过较少与我们实验室匹配的方向。
- Jian Tang homepage(https://sites.google.com/site/pkujiantang/). Related research direction: topic modeling.

References

References I

[1] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena.
Deepwalk: Online learning of social representations.
In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

[2] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang.
Heterogeneous network embedding via deep architectures.
In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 119–128. ACM, 2015.

References II

[3] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei.

Line: Large-scale information network embedding.

In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077. ACM, 2015.

[4] Aditya Grover and Jure Leskovec.

node2vec: Scalable feature learning for networks.

In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864. ACM, 2016.

References III

[5] Daixin Wang, Peng Cui, and Wenwu Zhu.
Structural deep network embedding.
In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1225–1234. ACM, 2016.

[6] Xiao Huang, Jundong Li, and Xia Hu.
Label informed attributed network embedding.
In *Proceedings of 10th ACM International Conference on Web Search and Data Mining (WSDM)*, 2017.

Other Papers I

- DeepWalk: Online Learning of Social Representations (by Bryan Perozzi et. al)
- GraRep: Learning Graph Representations with Global Structural Information (by Shaosheng Cao et. al)
- Network Representation Learning with Rich Text Information (by Cheng Yang et. al)
- LINE: Large-scale Information Network Embedding (by Jian Tang et. al)
- PTE: Predictive Text Embedding through Large-scale Heterogeneous Text Networks (by Jian Tang et. al)
- Heterogeneous Information Network Embedding for Meta Path based Proximity (by Zhipeng Huang et. al)

Other Papers II

- Heterogeneous Network Embedding via Deep Architectures (by Shiyu Chang et. al)
- Label Informed Attributed Network Embedding (by Xiao Huang et. al)
- Incorporate Group Information to Enhance Network Embedding (by Jifan Chen et. al)
- Community-enhanced Network Representation Learning for Network Analysis (by Cunchao Tu et. al)
- Tri-Party Deep Network Representation (by Shirui Pan)
- Asymmetric Transitivity Preserving Graph Embedding (by Mingdong Ou et. al)
- Structural Deep Network Embedding (by Daixin Wang et. al)

Other Papers III

- Cauchy Graph Embedding (by Dijun Luo et. al)
- Distributed Representations of Words and Phrases and their Compositionality (by Tomas Mikolov et. al)
- (Slides)Part III: Network Embedding with Deep Learning (by Xin Zhao)
- (Slides)Network Representation Learning: A Revisit in the Big Data Era (by Peng Cui)