Network Embedding

Hongwei Wang

04/26/2017

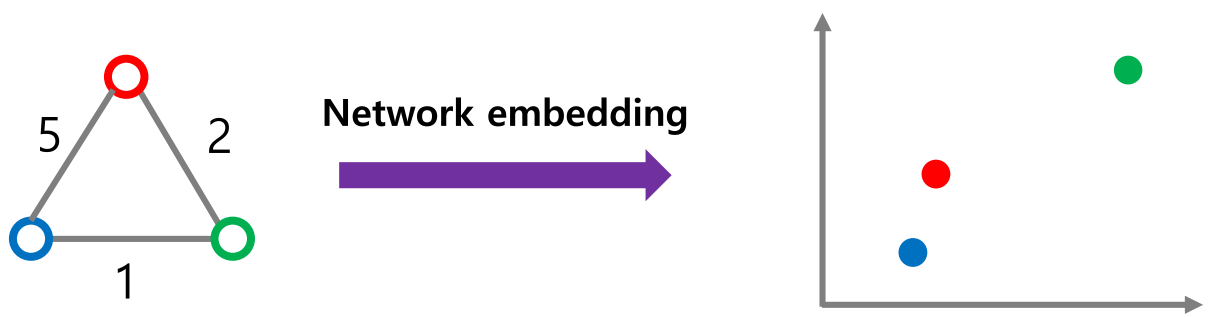
Introduction

- **Network embedding** tries to embed each node of a network into a low-dimensional vector space, which preserves the structural similarities or distances among the nodes in original network

- Network embedding can be viewed as a **dimension reduction** technology

- Also called **graph embedding** / **graph representation learning** / **graph feature learning**

- Potentially useful for **node classification**, **link prediction**, **clustering**, **recommender systems**, **anomaly detection**, **social network analysis**, **knowledge base**, etc. (In fact any task on network-structured data can benefit from network embedding)



Have a glimpse

long long ago

- PCA
- LDA
- MDS

not long ago

- Isomap
  Science, 2000
- LLE
  Science, 2000
- LE
  NIPS, 2001

recent 5 years

- Word2vec
  NIPS, 2013
- DeepWalk
  KDD, 2014
- LINE
  WWW, 2015
- TransR
  AAAI, 2015
- Node2vec
  KDD, 2016
- APP
  AAAI, 2017

- GraphEncoder
  AAAI, 2014
- HNE
  KDD, 2015
- SDNE
  KDD, 2016

- GraRep
  CIKM, 2015
- HOPE
  KDD, 2016
- LANE
  WSDM, 2017

Classical Dimension Reduction Methods

PCA

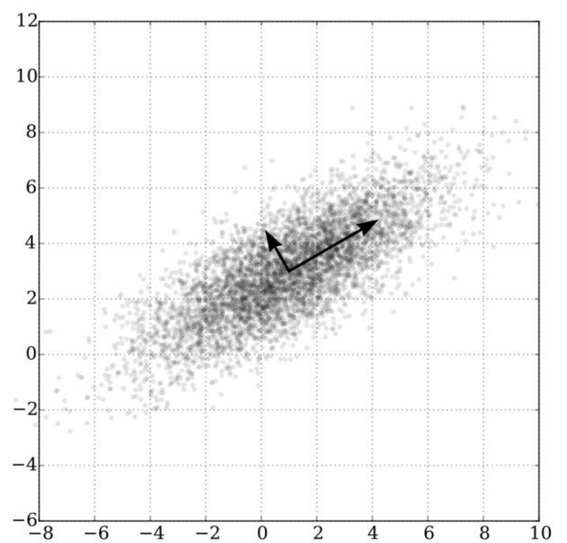
Principle Component Analysis

Karl Pearson

PCA

Motivation

- Given n d-dimensional samples $X^{d \times n} = \{x_1, x_2, \dots, x_n\}$, PCA seeks to find $d'(d' \ll d)$ orthogonal transformations $W^{d \times d'} = \{w_1, w_2, \dots, w_{d'}\}$, so that $W^T X$ **has the largest variance** (i.e., most separable).



PCA

Details

输入: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$;
低维空间维数 d'.

过程:

1: 对所有样本进行中心化: $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i - \frac{1}{m}\sum_{i=1}^{m}\boldsymbol{x}_i$;
2: 计算样本的协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$;
3: 对协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 做特征值分解;
4: 取最大的 d' 个特征值所对应的特征向量 $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'}$.

输出: 投影矩阵 $\mathbf{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'})$.

——《机器学习》，周志华，p.231

LDA

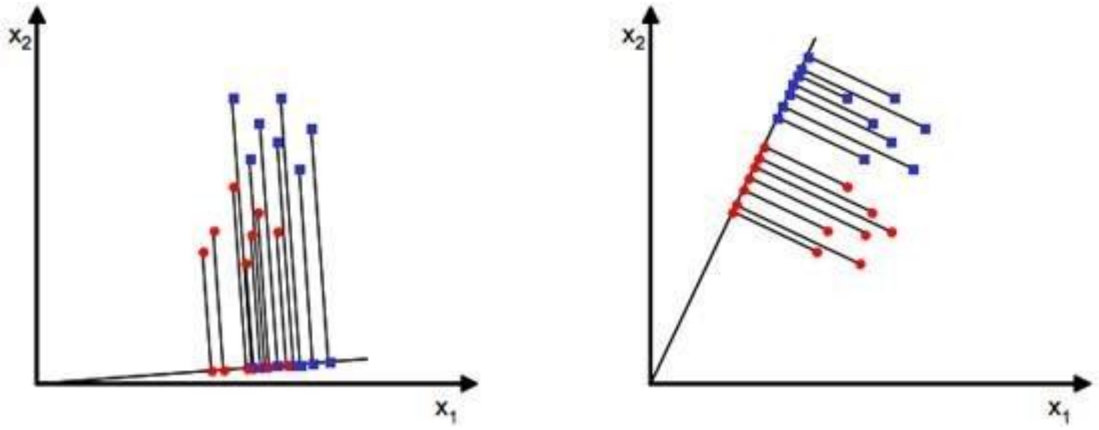
Linear Discriminant Analysis

Fisher

LDA

Motivation

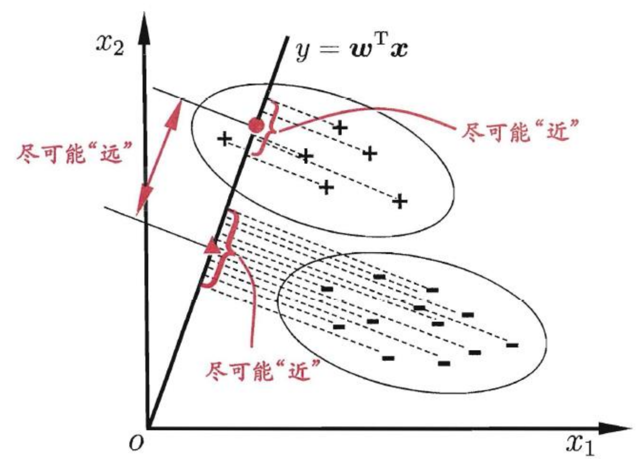
- LDA explicitly models the distance between and within the classes of data



LDA

Details

- Suppose binary classification
- $D = \{(x_i, y_i)\}$, μ_i: mean of data of the i-th class, Σ_i: covariance matrix of data of the i-th class
- make projected covariance matrix as small as possible, while make projected distance between the mean of two classes as large as possible



- maximize

$$J = \frac{\|\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\mu}_0 - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{\mu}_1\|_2^2}{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\Sigma}_0\boldsymbol{w} + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{\Sigma}_1\boldsymbol{w}} = \frac{\boldsymbol{w}^{\mathrm{T}}(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^{\mathrm{T}}\boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}}(\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1)\boldsymbol{w}}$$

MDS

Multiple Dimensional Scaling

Trevor F. Cox, Michael A. A. Cox

MDS

Motivation

- Given distance matrix $D = (d_{ij}) \in R^{m \times m}$, MDS seeks to find $z_1, z_2, \dots, z_m \in R^{d'} (d' \ll d)$, so that

$$\| z_i - z_j \| \approx d_{ij} \text{ as close as possible}$$

- MDS aims to place each object in *d'*-dimensional space such that the **between-object distances are preserved** as well as possible

MDS

Details

假定 m 个样本在原始空间的距离矩阵为 $\mathbf{D} \in \mathbb{R}^{m \times m}$, 其第 i 行 j 列的元素 $dist_{ij}$ 为样本 $\boldsymbol{x}_i$ 到 $\boldsymbol{x}_j$ 的距离. 我们的目标是获得样本在 d' 维空间的表示 $\mathbf{Z} \in \mathbb{R}^{d' \times m}$, $d' \leqslant d$, 且任意两个样本在 d' 维空间中的欧氏距离等于原始空间中的距离, 即 $\|\boldsymbol{z}_i - \boldsymbol{z}_j\| = dist_{ij}$.

令 $\mathbf{B} = \mathbf{Z}^{\mathrm{T}}\mathbf{Z} \in \mathbb{R}^{m \times m}$, 其中 $\mathbf{B}$ 为降维后样本的内积矩阵, $b_{ij} = \boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{z}_j$, 有

$$\begin{aligned} dist_{ij}^2 &= \|\boldsymbol{z}_i\|^2 + \|\boldsymbol{z}_j\|^2 - 2\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{z}_j \\ &= b_{ii} + b_{jj} - 2b_{ij} \ . \end{aligned} \tag{10.3}$$

为便于讨论, 令降维后的样本 $\mathbf{Z}$ 被中心化, 即 $\sum_{i=1}^m \boldsymbol{z}_i = \mathbf{0}$. 显然, 矩阵 $\mathbf{B}$ 的行与列之和均为零, 即 $\sum_{i=1}^m b_{ij} = \sum_{j=1}^m b_{ij} = 0$. 易知

$$\sum_{i=1}^m dist_{ij}^2 = \mathrm{tr}(\mathbf{B}) + m b_{jj} \ , \tag{10.4}$$

$$\sum_{j=1}^m dist_{ij}^2 = \mathrm{tr}(\mathbf{B}) + m b_{ii} \ , \tag{10.5}$$

$$\sum_{i=1}^m \sum_{j=1}^m dist_{ij}^2 = 2m\, \mathrm{tr}(\mathbf{B}) \ , \tag{10.6}$$

其中 $\mathrm{tr}(\cdot)$ 表示矩阵的迹(trace), $\mathrm{tr}(\mathbf{B}) = \sum_{i=1}^m \|\boldsymbol{z}_i\|^2$. 令

$$dist_{i\cdot}^2 = \frac{1}{m} \sum_{j=1}^m dist_{ij}^2 \ , \tag{10.7}$$

$$dist_{\cdot j}^2 = \frac{1}{m} \sum_{i=1}^m dist_{ij}^2 \ , \tag{10.8}$$

$$dist_{\cdot\cdot}^2 = \frac{1}{m^2} \sum_{i=1}^m \sum_{j=1}^m dist_{ij}^2 \ , \tag{10.9}$$

MDS

Details

由式(10.3)和式(10.4)~(10.9)可得

$$b_{ij} = -\frac{1}{2}(dist_{ij}^2 - dist_{i\cdot}^2 - dist_{\cdot j}^2 + dist_{\cdot\cdot}^2) \ , \tag{10.10}$$

由此即可通过降维前后保持不变的距离矩阵 $\mathbf{D}$ 求取内积矩阵 $\mathbf{B}$.

对矩阵 $\mathbf{B}$ 做特征值分解(eigenvalue decomposition), $\mathbf{B} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{T}}$, 其中 $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_d)$ 为特征值构成的对角矩阵, $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_d$, $\mathbf{V}$ 为特征向量矩阵. 假定其中有 d^* 个非零特征值, 它们构成对角矩阵 $\mathbf{\Lambda}_* = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_{d^*})$, 令 $\mathbf{V}_*$ 表示相应的特征向量矩阵, 则 $\mathbf{Z}$ 可表达为

$$\mathbf{Z} = \mathbf{\Lambda}_*^{1/2}\mathbf{V}_*^{\mathrm{T}} \in \mathbb{R}^{d^* \times m} \ . \tag{10.11}$$

在现实应用中为了有效降维, 往往仅需降维后的距离与原始空间中的距离尽可能接近, 而不必严格相等. 此时可取 $d' \ll d$ 个最大特征值构成对角矩阵 $\tilde{\mathbf{\Lambda}} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_{d'})$, 令 $\tilde{\mathbf{V}}$ 表示相应的特征向量矩阵, 则 $\mathbf{Z}$ 可表达为

$$\mathbf{Z} = \tilde{\mathbf{\Lambda}}^{1/2}\tilde{\mathbf{V}}^{\mathrm{T}} \in \mathbb{R}^{d' \times m} \ . \tag{10.12}$$

——《机器学习》，周志华，p.227-229

Classical Embedding Methods

Isomap (Science '00)

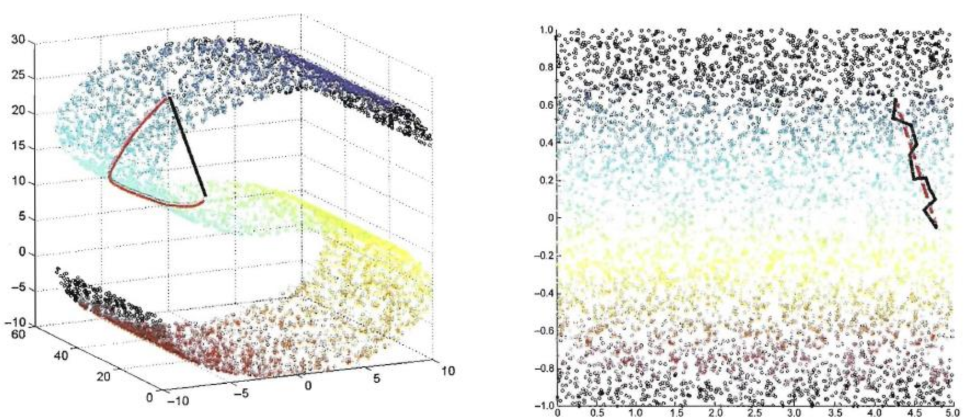
Isometric Mapping

J. B. Tenenbaum, V De Silva, JC Langford
Science, 2000

Isomap (Science '00)

Motivation

- Isomap provides a simple method for estimating **intrinsic geometry manifold** based on a rough estimate of each data point's **neighbors** on the manifold.



(a) 测地线距离与高维直线距离　　(b) 测地线距离与近邻距离

Isomap (Science '00)

Details

输入: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$;
近邻参数 k;
低维空间维数 d'.

过程:

for $i = 1, 2, \ldots, m$ **do**
  确定 $\boldsymbol{x}_i$ 的 k 近邻;
  $\boldsymbol{x}_i$ 与 k 近邻点之间的距离设置为欧氏距离, 与其他点的距离设置为无穷大;
end for
调用最短路径算法计算任意两样本点之间的距离 $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)$;
将 $\mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ 作为 MDS 算法的输入;
return MDS 算法的输出

输出: 样本集 D 在低维空间的投影 $Z = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_m\}$.

——《机器学习》，周志华，p.235

LLE (Science '00)

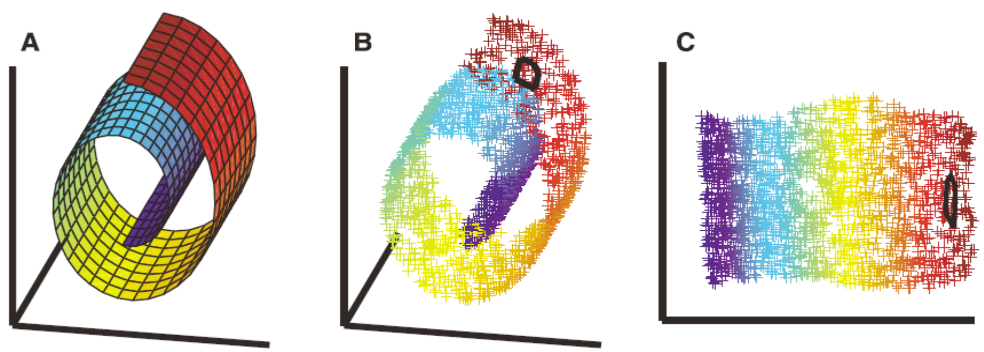
Locally Linear Embedding

Sam T. Roweis, Lawrence K. Saul
Science, 2000

LLE (Science '00)

Introduction

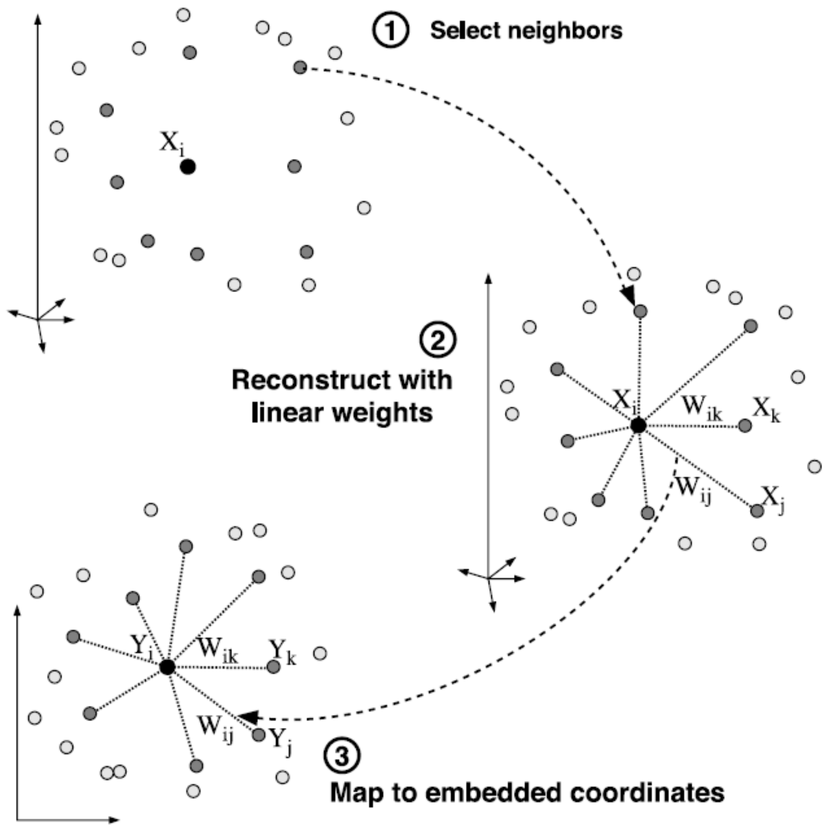
- An unsupervised learning algorithm that computes low-dimensional, **neighborhood-preserving** embeddings of high-dimensional inputs



- What is the difference between LLE and Isomap?
  - Isomap keeps **distances** between local instances;
  - LLE keeps **linear dependency** between local instances.

LLE (Science '00)

Details



LLE (Science '00)

Details

Reconstruction step

Given data set $\{X_i\}$, fin

$$\varepsilon(W) = \sum_i \left| \vec{X}_i - \Sigma_j W_{ij} \vec{X}_j \right|^2$$

subject to:

- Each data point is reconstructed only from its **k**-nearest neighbors (only a fraction of **W** is non-zero)
- The rows of **W** sum to one ($\sum_j W_{ij} = 1$)

Embedding step

Given **W**, find **Y** to minimize

$$\Phi(Y) = \sum_i \left| \vec{Y}_i - \Sigma_j W_{ij} \vec{Y}_j \right|^2$$

Y is the embedding result.

LE (NIPS '01)

Laplacian Eigenmaps

Mikhail Belkin, Partha Viyogi

NIPS, 2001

LE (NIPS '01)

Motivation

- **Problem:** Given a set $(\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_k})$ in R^d, find a set of points $(\boldsymbol{y_1}, \boldsymbol{y_2}, \ldots, \boldsymbol{y_k})$ in $R^{d'}$ $(d' \ll d)$ such that $\boldsymbol{y_i}$ represents $\boldsymbol{x_i}$.

- **Minimization Problem**

$$\min \sum_{ij} W_{ij} \parallel \boldsymbol{y_i} - \boldsymbol{y_j} \parallel^2$$

LE (NIPS '01)

Motivation

- **Minimization Problem**

$$\min \sum\nolimits_{ij} W_{ij} \parallel \boldsymbol{y_i} - \boldsymbol{y_j} \parallel^2$$

which is equivalent to

$$\begin{aligned} &\arg\min_{\mathbf{y}} \ \ \mathrm{tr}(Y^{\mathrm{T}} L Y) \\ &\mathrm{s.t.} \ \ Y^T D Y = 1 \end{aligned}$$

- **Solution**
  - Construct the adjacency graph
  - Compute the weights W_{ij}
  - Compute Laplacian matrix $L = D - W$, where $D_{ii} = \sum_j W_{ij}$ is diagonal matrix
  - Compute eigenvalues and eigenvectors of the generalized eigenvector problem: $Lf = \lambda Df$, and the d' eigenvectors corresponding to the d' smallest eigenvalues except 0 are taken as embeddings

Word2vec-like Network Embedding Methods

Word2vec (NIPS '13)

Distributed Representation of Words and Phrases and their Compositionality

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean
NIPS, 2013

Word2vec (NIPS '13)

Introduction

- **The Skip-Gram Model**
  - The training objective is to find word representations that are useful for **predicting the surrounding words** in a sentence
  - I.e., to maximize

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c,j\neq 0}\log p(w_{t+j}|w_t)$$

where

$$p(w_O|w_I)=\frac{\exp\left({v'_{w_O}}^{\top}v_{w_I}\right)}{\sum_{w=1}^{W}\exp\left({v'_w}^{\top}v_{w_I}\right)}$$

  - The above objective function is computationally intractable

Word2vec (NIPS '13)

Introduction

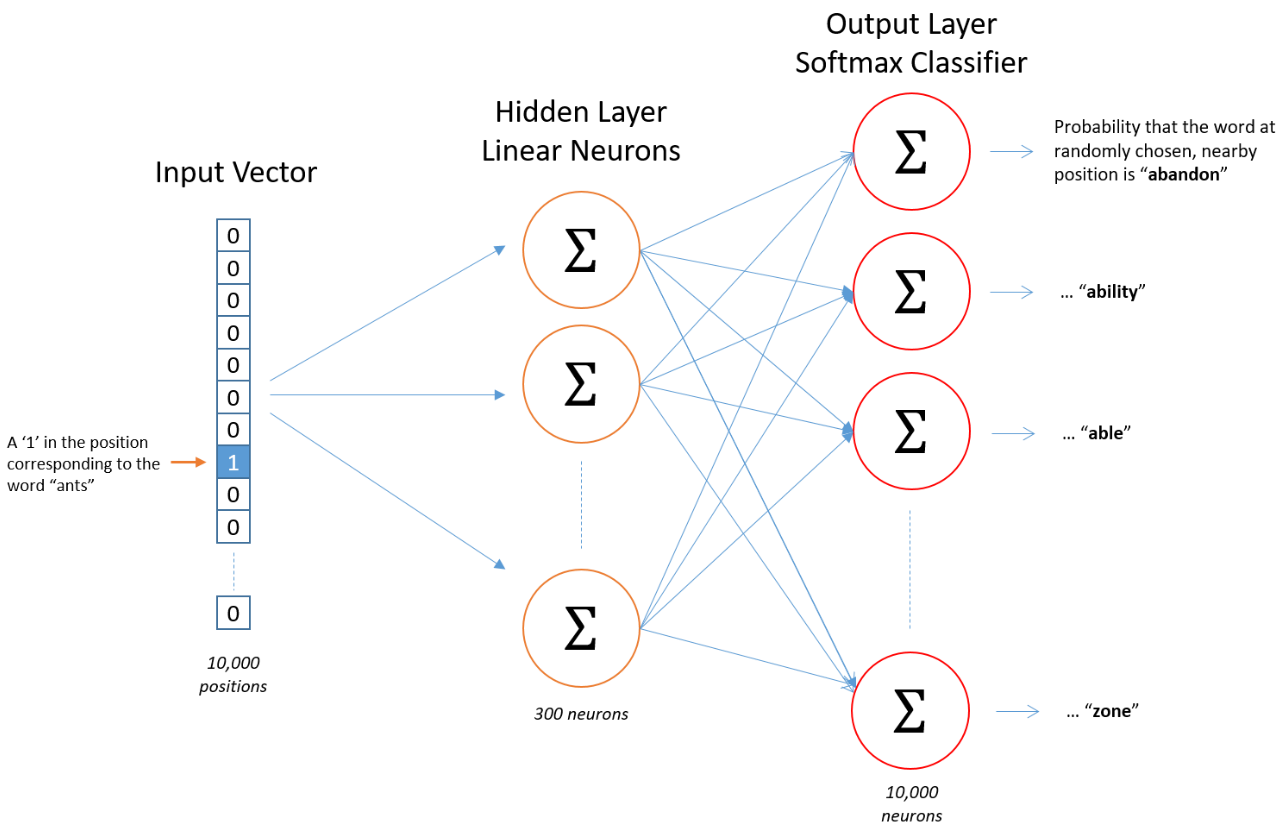
- **The Skip-Gram Model**
  - An alternative to the full softmax is **Negative Sampling**:

$$\log \sigma({v'_{w_O}}^\top v_{w_I}) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)} \left[\log \sigma(-{v'_{w_i}}^\top v_{w_I}) \right]$$

  - *k* in the range 5-20 are useful for small training datasets, while for large datasets the *k* can be as small as 2-5

Word2vec (NIPS '13)

From perspective of neural network



DeepWalk (KDD '14)

DeepWalk: Online Learning of Social Representations

Bryan Perozzi, Rami Al-Rfou, Steven Skiena

KDD, 2014

DeepWalk (KDD '14)

Introduction

- **Random walk + Word2vec**
  - A walk samples **uniformly** from the neighbors of the last vertex visited

$$\underset{\Phi}{\text{minimize}} \quad -\log \Pr\left(\{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid \Phi(v_i)\right)$$

LINE (WWW '15)

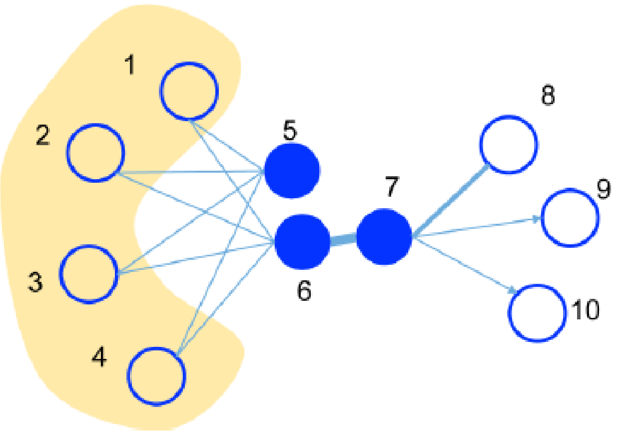
LINE: Large-scale Information Network Embedding

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, Qiaozhu Mei
WWW, 2015

LINE (WWW '15)

Proximity

- First-order proximity
- Second-order proximity



LINE (WWW '15)

First-order proximity

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

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

$$\hat{p}_1(i,j) = \frac{w_{ij}}{W}$$

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

Use negative sampling to optimize the objective function to avoid trivial solution $u_{ik} = \infty$

LINE (WWW '15)

Second-order proximity

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

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

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

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

TransR (AAAI '15)

Learning Entity and Relation Embeddings for Knowledge Graph Completion

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, Xuan Zhu

AAAI, 2015

TransR (AAAI '15)

Introduction

- Embed **knowledge graph** into a continuous vector space while preserving certain information
- The difference with general network embedding lies in that:
  - Nodes in knowledge graphs are entities with different types
  - Edges in knowledge graphs are relations of different types
- Prior work TransE (NIPS 13):
  - Ensures $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ when (h, r, t) holds
  - Has issues when modeling 1-to-N, N-to-1, and N-to-N relations
- Prior work TransH (AAAI 14):
  - Ensures $\mathbf{h}_\perp + \mathbf{r} \approx \mathbf{t}_\perp$ when (h, r, t) holds, where $\mathbf{h}_\perp = \mathbf{h} - \mathbf{w}_r^{\mathrm{T}}\mathbf{h}\mathbf{w}_r$ and $\mathbf{t}_\perp = \mathbf{t} - \mathbf{w}_r^{\mathrm{T}}\mathbf{t}\mathbf{w}_r$
  - Still embeds entities and relations in the same space

TransR (AAAI '15)

Details

- In TransR, for each triple (h, r, t), entities embeddings are $\mathbf{h}, \mathbf{t} \in \mathbb{R}^k$ and relation embeddings are $\mathbf{r} \in \mathbb{R}^d$.

- Score function: $f_r(h, t) = \|\mathbf{h}\mathbf{M}_r + \boldsymbol{r} - \mathbf{t}\mathbf{M}_r\|_2^2$, where $\mathbf{M}_r$ is the projection matrix for relation r.

Node2vec (KDD '16)

node2vec: Scalable Feature Learning for Networks

Aditya Grover, Jure Leskovec
KDD, 2016

Node2vec (KDD '16)

Introduction

- **Random walk + Word2vec**
  - Then what's the difference between node2vec and DeepWalk?

- **Sampling strategy**

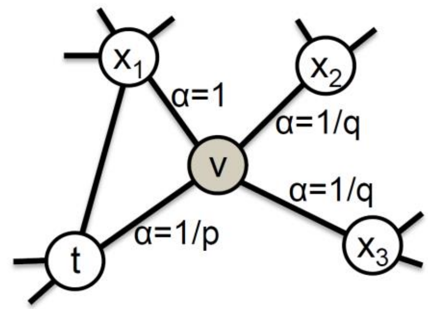
$$P(c_i = x \mid c_{i-1} = v) = \begin{cases} \frac{\pi_{vx}}{Z} & \text{if } (v, x) \in E \\ 0 & \text{otherwise} \end{cases}$$

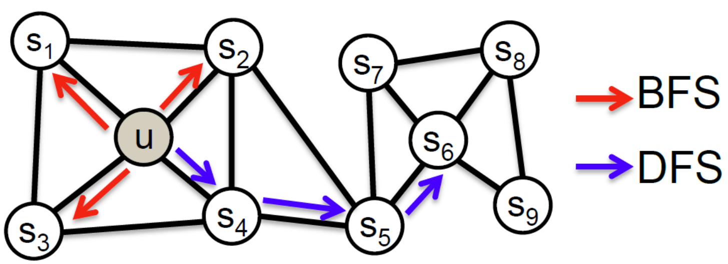
$$\pi_{vx} = \alpha_{pq}(t, x) \cdot w_{vx}$$

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

Node2vec (KDD '16)

Introduction





APP (AAAI '17)

Scalable Graph Embedding for Asymmetric Proximity

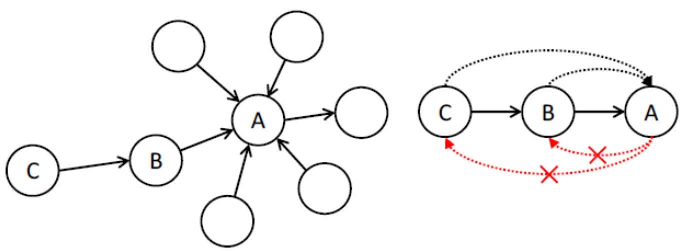
Chang Zhou, Yuqiong Liu, Xiaofei Liu, Zhongyi Liu, Jun Gao

AAAI, 2017

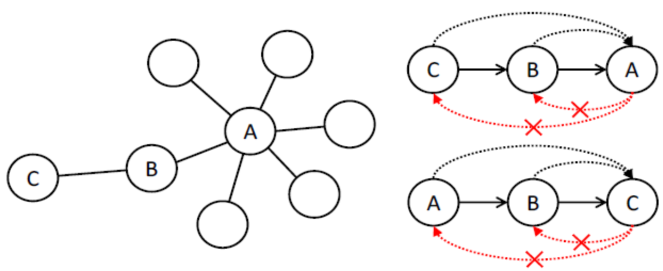
APP (AAAI '17)

Motivation

- **DeepWalk, LINE, node2vec only consider symmetric proximities**
  - The paths sampled by DeepWalk and node2vec are treated as a word sequence, in which words (nodes) are symmetric
  - Insufficient in manly applications, even undirected graphs



(a) Directed



(b) Undirected

APP (AAAI '17)

Motivation

- The prediction probability

$$p(v|u) = \frac{\exp(\vec{s_u} \cdot \vec{t_v})}{\sum_{n \in V} \exp(\vec{s_u} \cdot \vec{t_n})}$$

- Objective function

$$log\sigma(\vec{s_u} \cdot \vec{t_v}) + k \cdot E_{t_n \sim P_D}[log\sigma(-\vec{s_u} \cdot \vec{t_n})]$$

Autoencoder-based Network Embedding Methods

GraphEncoder (AAAI '14)

Learning Deep Representations for Graph Clustering

Fei Tian, Bin Gao, Qing Cui, Enhong Chen, Tie-Yan Liu
AAAI, 2014

GraphEncoder (AAAI '14)

Introduction

- **Sparse Autoencoder + K-means**
  - Input: normalized similarity matrix of graph
  - Train autoencoder layer by layer
  - Loss function of autoencoder:

$$Loss(\theta) = \sum_{i=1}^{n} ||y_i - x_i||_2 + \beta KL(\rho||\hat{\rho})$$

where the second term is the sparsity penalty

HNE (KDD '15)

Heterogeneous Network Embedding via Deep Architectures

Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C. Aggarwal, Thomas S. Huang

KDD, 2015

HNE (KDD '15)

Motivation

- **Consider a network with different types of nodes**
  - Online media platform with images and texts
- **HNE tries to embed different types of nodes into a unified representation space**

HNE (KDD '15)

Details (linear version)

- suppose there are two types of nodes in network: image and text
- Loss function for image-image proximity:

$$L(\boldsymbol{x}_i, \boldsymbol{x}_j) = \log\left(1 + \exp\left(-\boldsymbol{A}_{i,j} d(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)\right)$$

where **A** is edge indicator, and

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = s(\boldsymbol{x}_i, \boldsymbol{x}_j) - t_{II}$$

$$s(\boldsymbol{x_i}, \boldsymbol{x_j}) = \tilde{\boldsymbol{x}_i}^T \tilde{\boldsymbol{x}_j} = (\boldsymbol{U}^T \boldsymbol{x}_i)^T \boldsymbol{U}^T \boldsymbol{x}_j = \boldsymbol{x_i}^T \boldsymbol{M}_{II} \boldsymbol{x_j}$$

- Loss function for the whole model:

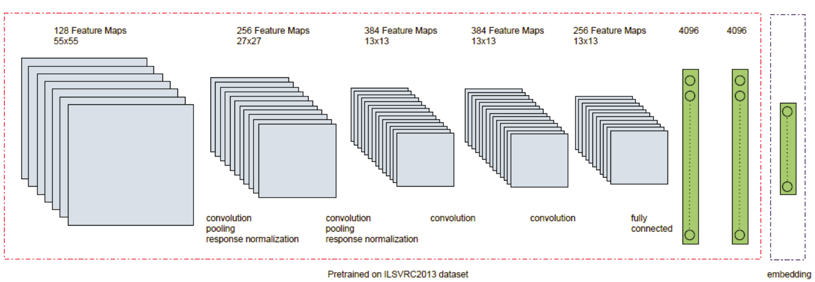
$$\min_{\boldsymbol{U},\boldsymbol{V}} \frac{1}{N_{II}} \sum_{v_i, v_j \in \mathcal{V}_I} L(\boldsymbol{x}_i, \boldsymbol{x}_j) + \frac{\lambda_1}{N_{TT}} \sum_{v_i, v_j \in \mathcal{V}_T} L(\boldsymbol{z}_i, \boldsymbol{z}_j)$$
$$+ \frac{\lambda_2}{N_{IT}} \sum_{v_i \in \mathcal{V}_I, v_j \in \mathcal{V}_T} L(\boldsymbol{x}_i, \boldsymbol{z}_j) + \lambda_3(\|\boldsymbol{U}\|_F^2 + \|\boldsymbol{V}\|_F^2),$$

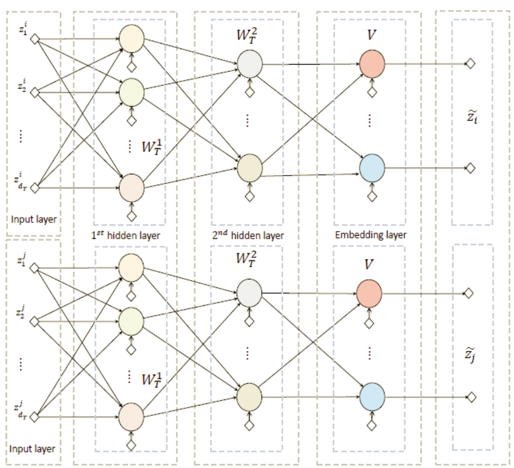
HNE (KDD '15)

Details (deep version)

- Loss function for the whole model:

$$\begin{aligned}
\min_{\mathcal{D}'_I,\mathcal{D}'_T} \quad & \frac{1}{N_{II}} \sum_{v_i,v_j\in\mathcal{V}_I} L'(\tilde{p}_{\mathcal{D}'_I}(\boldsymbol{X}_i),\tilde{p}_{\mathcal{D}'_I}(\boldsymbol{X}_j)) \\
& + \frac{\lambda_1}{N_{TT}} \sum_{v_i,v_j\in\mathcal{V}_T} L'(\tilde{q}_{\mathcal{D}'_T}(\boldsymbol{z}_i),\tilde{q}_{\mathcal{D}'_T}(\boldsymbol{z}_j)) \\
& + \frac{\lambda_2}{N_{IT}} \sum_{v_i\in\mathcal{V}_I,v_j\in\mathcal{V}_T} L'(\tilde{p}_{\mathcal{D}'_I}(\boldsymbol{X}_i),\tilde{q}_{\mathcal{D}'_T}(\boldsymbol{z}_j))
\end{aligned}$$





SDNE (KDD '16)

Structural Deep Network Embedding

Daixin Wang, Peng Cui, Wenwu Zhu
KDD, 2016

SDNE (KDD '16)

Loss function

- **Reconstruction loss term**

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

- **Proximity loss term**

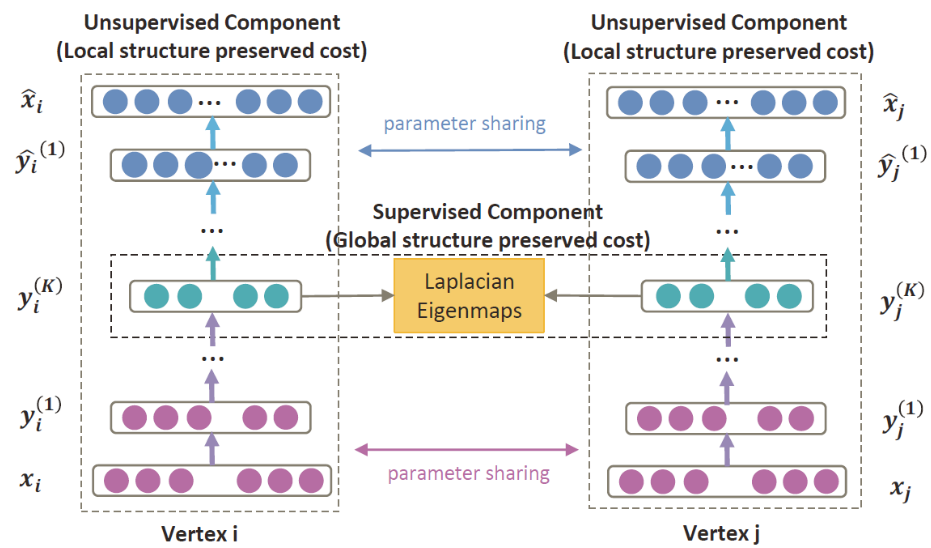
$$\mathcal{L}_{1st} = \sum_{i,j=1}^{n} s_{i,j} \|\mathbf{y}_i^{(K)} - \mathbf{y}_j^{(K)}\|_2^2$$
$$= \sum_{i,j=1}^{n} s_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|_2^2$$

- **Regularization term**

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

- **Loss function**

$$\mathcal{L}_{mix} = \mathcal{L}_{2nd} + \alpha \mathcal{L}_{1st} + \nu \mathcal{L}_{reg}$$



Matrix Factorization-based Network Embedding Methods

HOPE (KDD '16)

Asymmetric Transitivity Preserving Graph Embedding

Mingdong Ou, Peng Cui, Jian Pei, Wenwu Zhu
KDD, 2016

HOPE (KDD ‘16)

Introduction

- Existing graph embedding methods cannot preserve the **asymmetric** transitivity well, which depicts the correlation among **directed** edges.

- High-Order Proximity preserving Embedding (HOPE) is proposed, which is scalable to preserve high-order proximities of large scale graphs and capable of capturing the asymmetric transitivity

HOPE (KDD '16)

Details

- **G**={**V**, **E**}; **A**: adjacency matrix; **S**: high-order proximity matrix; $\mathbf{U}^s$: source embedding vectors; $\mathbf{U}^t$: target embedding vectors
- Loss function:

$$\min \left\| \mathbf{S} - \mathbf{U}^s \cdot {\mathbf{U}^t}^{\mathrm{T}} \right\|_F^2$$

- **S** shares a general formulation in many measurements:

$$\mathbf{S} = \mathbf{M}_{\boldsymbol{g}}^{-1} \cdot \mathbf{M}_{\boldsymbol{l}}$$

where

- In Katz Index: $\mathbf{M}_{\boldsymbol{g}} = \mathbf{I} - \beta \cdot \mathbf{A}$, $\mathbf{M}_{\boldsymbol{l}} = \beta \cdot \mathbf{A}$
- In Rooted PageRank: $\mathbf{M}_{\boldsymbol{g}} = \mathbf{I} - \boldsymbol{\alpha} \cdot \mathbf{P}$, $\mathbf{M}_{\boldsymbol{l}} = (1 - \alpha) \cdot \mathbf{I}$

……

LANE (WSDM '17)

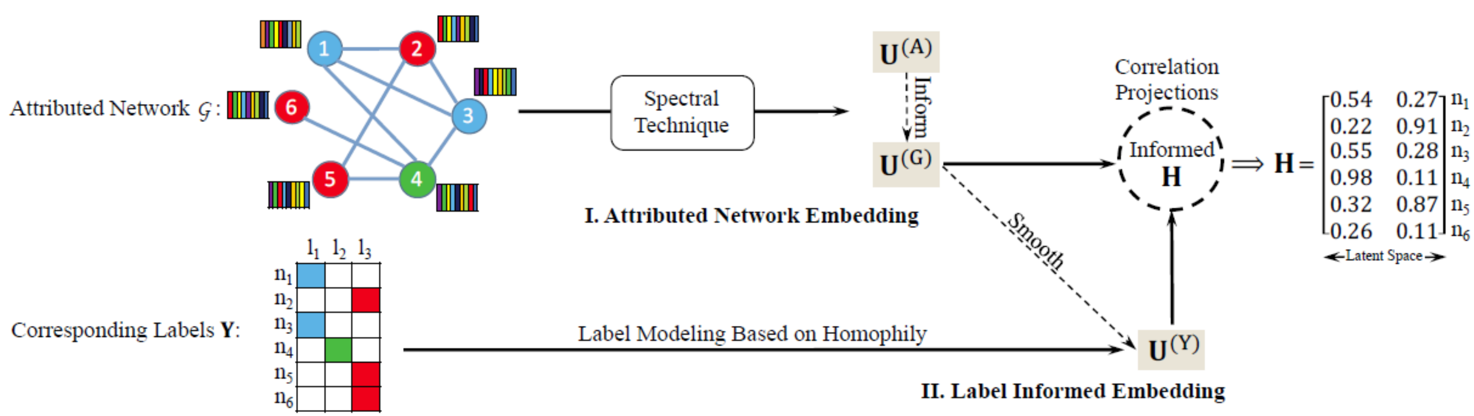
Label Informed Attributed Network Embedding

Xiao Huang, Jundong Li, Xia Hu
WSDM, 2017

LANE (WSDM '17)

Introduction

- Combine **structural (G)** information, **attribute (A)** information, and **label (Y)** information together
- Based on spectral graph theory (recall Laplacian Eigenmaps)



Network Embedding in RS

- In the above papers, **node classification** and **link prediction** task are most considered in experiment part
  - Node classification: classify nodes based on the features learned by network embedding
  - Link prediction: predict the presence of unobserved links

- Few experiments are for recommendation
  - APP: item recommendation in Taobao

- No prior work focuses on recommendation explicitly