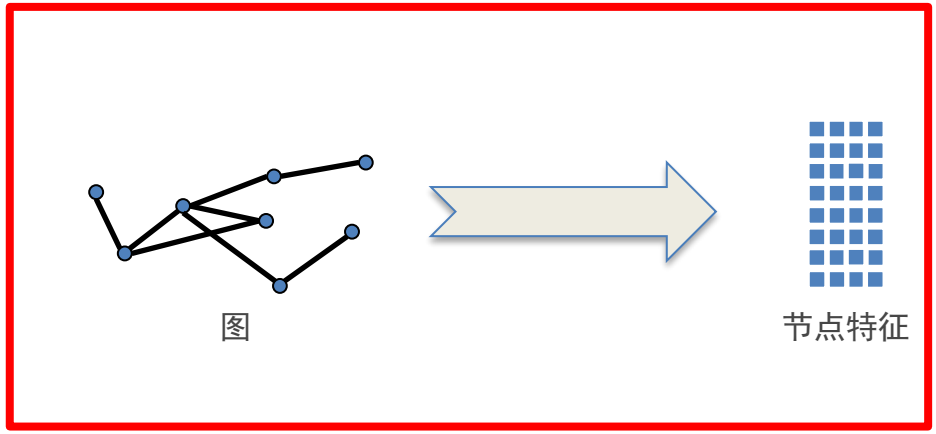
深蓝学院
shenlanxueyuan.com

图嵌入

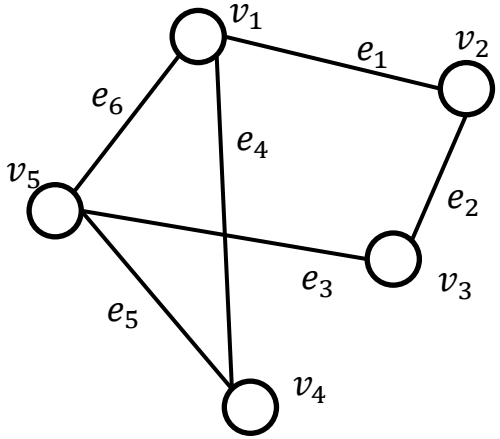
目录

图上的机器学习



图上的特征提取

邻接矩阵



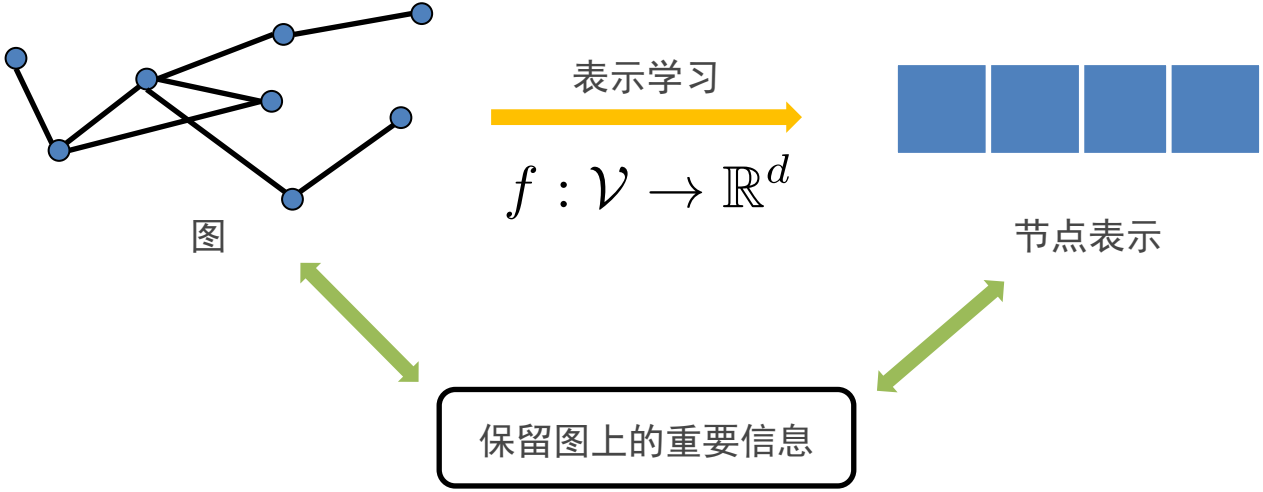
$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix}$$

稀疏

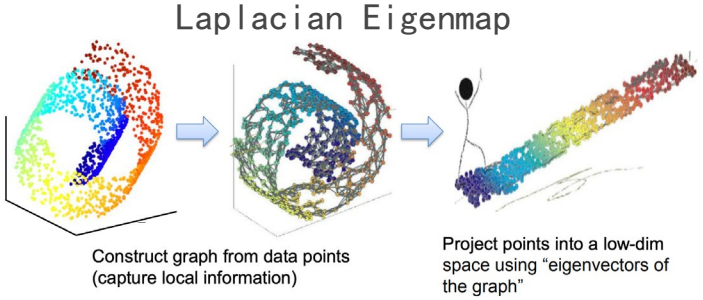
高维度

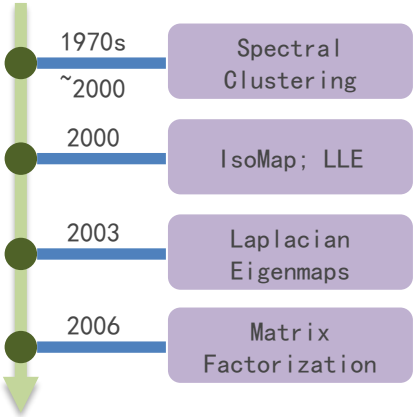
学习低维的节点表示

节点表示学习



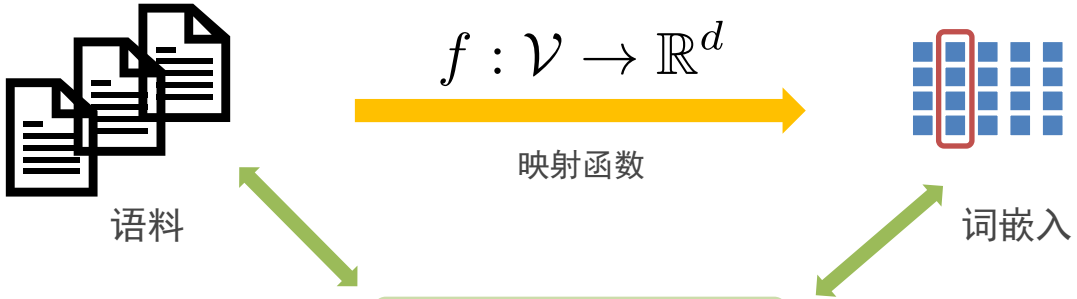
节点表示学习1.0：数据降维





- 通常具有很高的复杂度
- 无法应用到大规模的图上

word2vec词嵌入

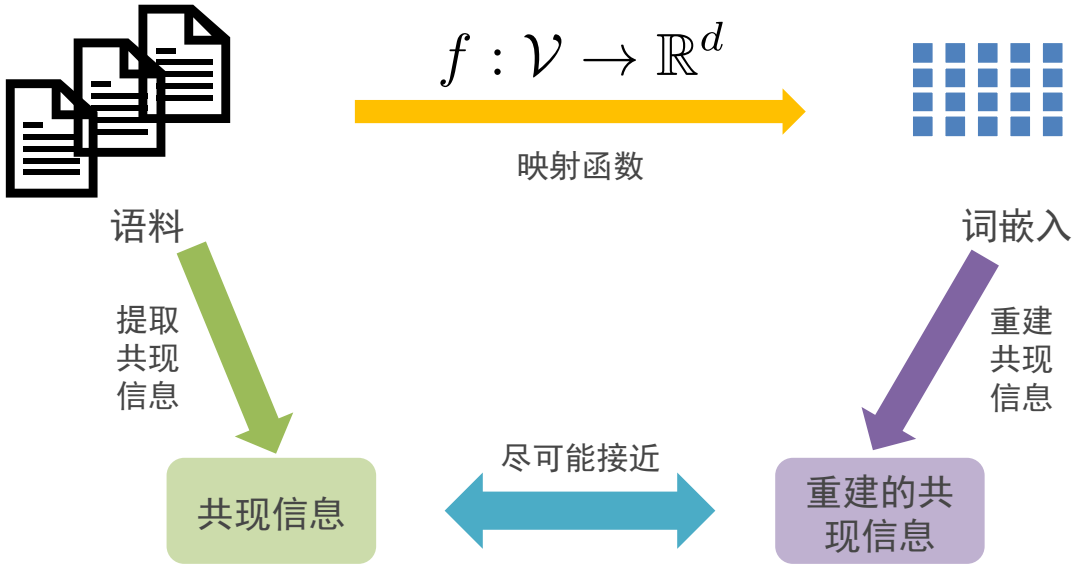


保留语义信息

“You shall know a word by the company it keeps” —J.R. Firth:11

共现信息十分重要

word2vec词嵌入



$$\max P(\{by, the, it, keeps\}|company)$$

“You shall know a word by the company it keeps” —J. R. Firth:11

节点表示学习2.0：图嵌入

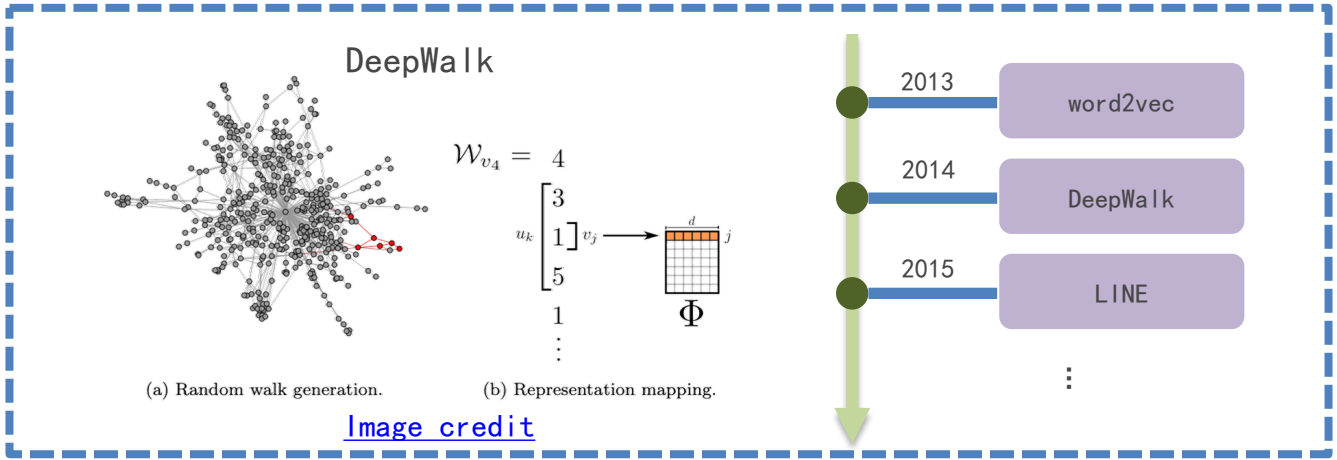
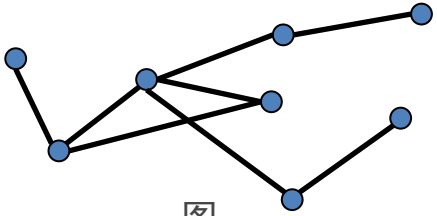


Image credit

- 通常比较高效
- 有较好的可拓展性

节点嵌入



图

表示学习

$$f : \mathcal{V} \rightarrow \mathbb{R}^d$$

节点嵌入

保留图上的重要信息

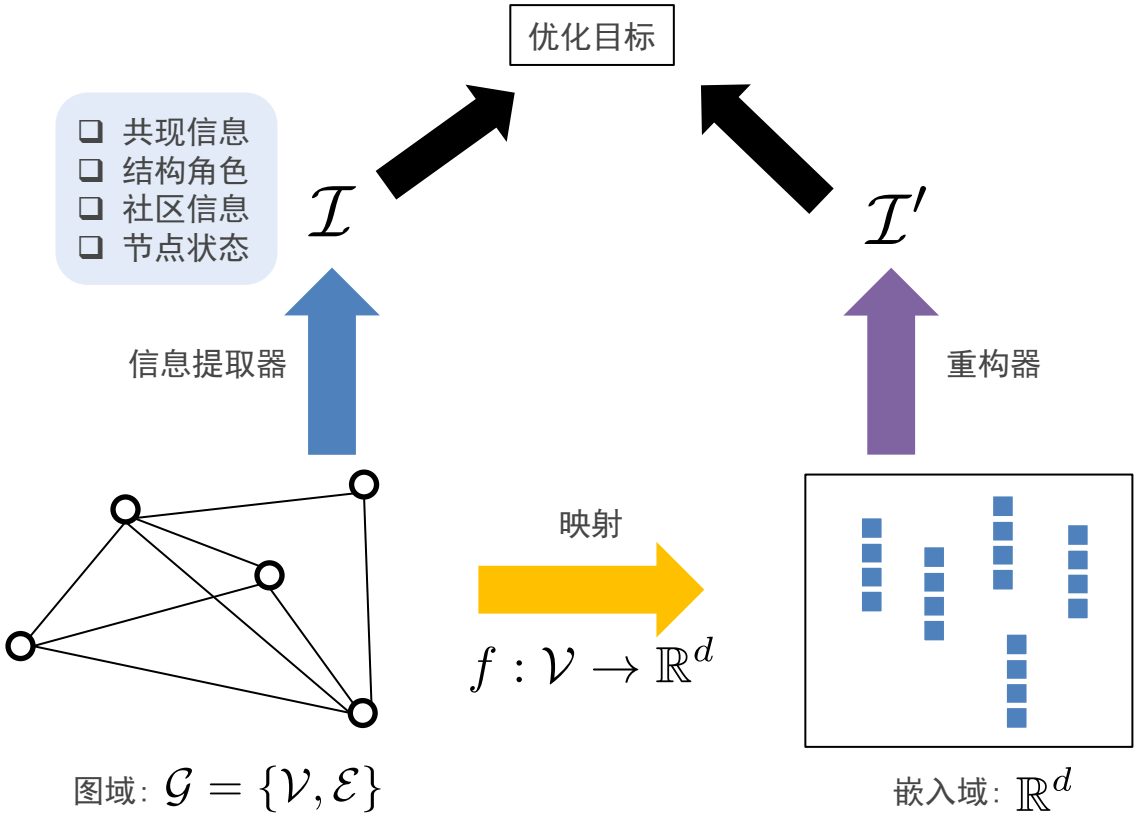
要保留什么信息？

- 共现信息
- 结构角色
- 社区信息
- 节点状态

如何保留这些信息？

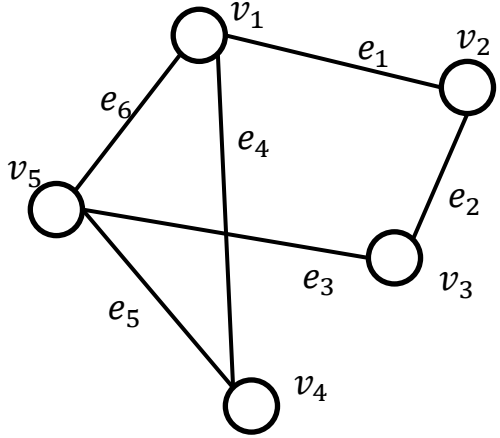
利用节点嵌入尽可能的重构要保留的信息

图嵌入的通用框架


优化目标
共现信息
结构角色
社区信息
节点状态
$\mathcal{I}$
$\mathcal{I}'$
信息提取器
重构器
映射
$f: \mathcal{V} \rightarrow \mathbb{R}^d$
图域: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$
嵌入域: $\mathbb{R}^d$

目录

简单图



无向

同质

节点和边上没有额外信息

目录

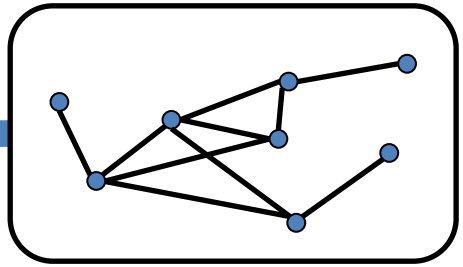
- 简单图嵌入
  - 保留邻域信息
  - 保留结构角色
  - 保留节点状态
  - 保留社区结构

保留共现信息的图嵌入：DeepWalk

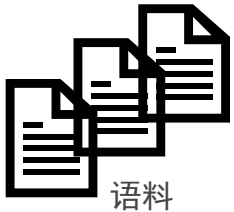
“虚拟语言”

- 词汇：节点
- 句子：随机游走

随机游走



图



语料

$f : \mathcal{V} \rightarrow \mathbb{R}^d$

映射函数

词嵌入

提取
共现
信息

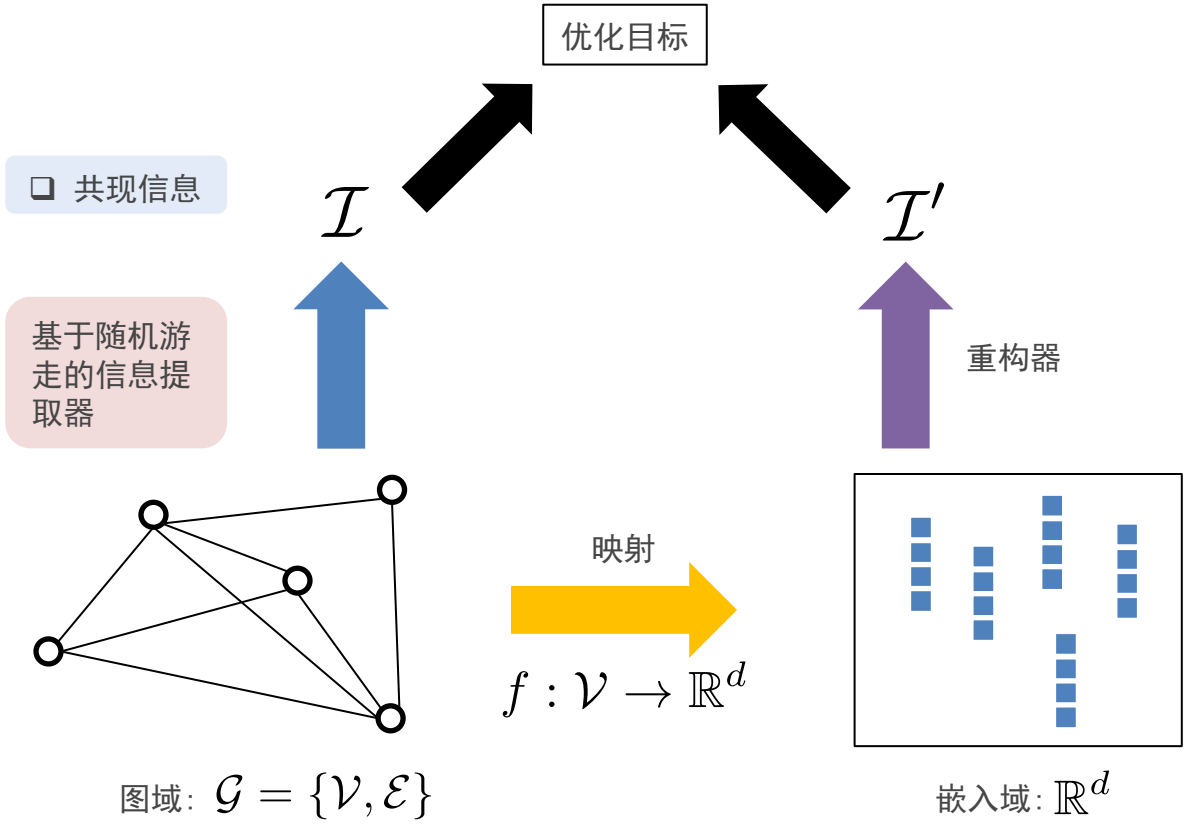
重建
共现
信息

共现信息

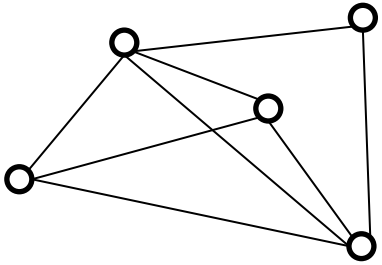
尽可能接近

重建的共现
信息

保留共现信息的图嵌入：DeepWalk

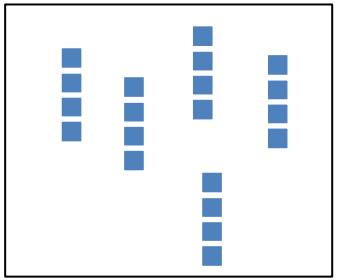


映射函数



映射

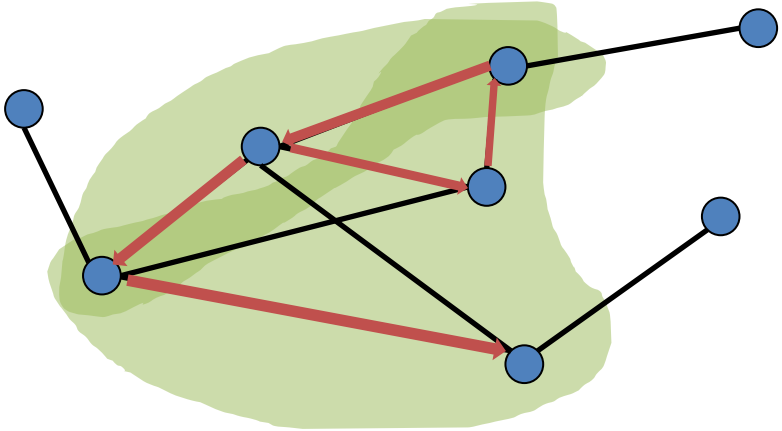
$$f : \mathcal{V} \rightarrow \mathbb{R}^d$$



$$f(v_i) = \mathbf{W}\mathbf{e}_i$$

$$f(v_2) = \mathbf{W} \times \mathbf{e}_2$$

随机游走



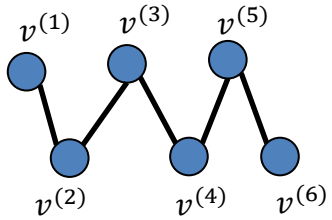
这样随机生成的途径是图上的一个随机游走

随机游走在点$v^{(t)}$根据如下均匀分布选择下一个节点

$$p\left(v^{(t+1)} \mid v^{(t)}\right)=\begin{cases}\frac{1}{d(v^{(t)})}, & \text{if } v^{(t+1)} \in \mathcal{N}\left(v^{(t)}\right) \\ 0, & \text{otherwise}\end{cases}$$

基于随机游走的信息提取器

上下文节点：中心节点一定距离内的点



中心节点	上下文节点
$v^{(1)}$	$v^{(2)}, v^{(3)}$
$v^{(2)}$	$v^{(1)}, v^{(3)}, v^{(4)}$
$v^{(3)}$	$v^{(1)}, v^{(2)}, v^{(4)}, v^{(5)}$
$v^{(4)}$	$v^{(2)}, v^{(3)}, v^{(5)}, v^{(6)}$
$v^{(5)}$	$v^{(3)}, v^{(4)}, v^{(6)}$
$v^{(6)}$	$v^{(4)}, v^{(5)}$

随机游走 $\mathcal{W} = (v^{(1)}, \ldots, v^{(6)})$

共现信息 $\mathcal{I}_{\mathcal{W}}$

- 从每个节点出发
- 重复多次

$$\mathcal{R} = \{\mathcal{W}\}$$

$$\mathcal{I} = \bigcup_{\mathcal{W} \in \mathcal{R}} \mathcal{I}_{\mathcal{W}}$$

重构器

中心节点	上下文节点
$v^{(1)}$	$v^{(2)}, v^{(3)}$
$v^{(2)}$	$v^{(1)}, v^{(3)}, v^{(4)}$
$v^{(3)}$	$v^{(1)}, v^{(2)}, v^{(4)}, v^{(5)}$
$v^{(4)}$	$v^{(2)}, v^{(3)}, v^{(5)}, v^{(6)}$
$v^{(5)}$	$v^{(3)}, v^{(4)}, v^{(6)}$
$v^{(6)}$	$v^{(4)}, v^{(5)}$

共现信息

$$\mathcal{I}_{\mathcal{W}} = \{(v_{cen}, v_{con})\}$$

$$\mathcal{I} = \bigcup_{\mathcal{W} \in \mathcal{R}} \mathcal{I}_{\mathcal{W}}$$

重构共现信息

利用节点嵌入预测出上下文节点

$$p(v_{con}|v_{cen})$$

$$\prod_{(v_{cen}, v_{con}) \in \mathcal{I}_{\mathcal{W}}} p(v_{con}|v_{cen})$$

重构器

中心节点	上下文节点
$v^{(1)}$	$v^{(2)}, v^{(3)}$
$v^{(2)}$	$v^{(1)}, v^{(3)}, v^{(4)}$
$v^{(3)}$	$v^{(1)}, v^{(2)}, v^{(4)}, v^{(5)}$
$v^{(4)}$	$v^{(2)}, v^{(3)}, v^{(5)}, v^{(6)}$
$v^{(5)}$	$v^{(3)}, v^{(4)}, v^{(6)}$
$v^{(6)}$	$v^{(4)}, v^{(5)}$

两个映射函数

$$f_{cen}(v_i) = \mathbf{W}_{cen}\mathbf{e}_i$$

$$f_{con}(v_i) = \mathbf{W}_{con}\mathbf{e}_i$$

重构共现信息

利用节点嵌入预测出上下文节点

$$p(v_{con}|v_{cen})$$

$$\frac{\exp\left(f_{\text{con}}\ (v_{\text{con}})^{\top} f_{\text{cen}}\ (v_{\text{cen}})\right)}{\sum_{v\in\mathcal{V}} \exp\left(f_{\text{con}}\ (v)^{\top} f_{\text{cen}}\ (v_{\text{cen}})\right)}$$

重构器

中心节点	上下文节点
$v^{(1)}$	$v^{(2)}, v^{(3)}$
$v^{(2)}$	$v^{(1)}, v^{(3)}, v^{(4)}$
$v^{(3)}$	$v^{(1)}, v^{(2)}, v^{(4)}, v^{(5)}$
$v^{(4)}$	$v^{(2)}, v^{(3)}, v^{(5)}, v^{(6)}$
$v^{(5)}$	$v^{(3)}, v^{(4)}, v^{(6)}$
$v^{(6)}$	$v^{(4)}, v^{(5)}$

共现信息

$$\mathcal{I}_{\mathcal{W}} = \{(v_{cen}, v_{con})\}$$

$$\mathcal{I} = \bigcup_{\mathcal{W} \in \mathcal{R}} \mathcal{I}_{\mathcal{W}}$$

重构共现信息

利用节点嵌入预测出上下文节点

$$p(v_{con} | v_{cen})$$

$$\prod_{\mathcal{W} \in \mathcal{R}} \prod_{(v_{cen}, v_{con}) \in \mathcal{I}_{\mathcal{W}}} \frac{\exp\left(f_{\text{con}}\ (v_{\text{con}})^{\top} f_{\text{cen}}\ (v_{\text{cen}})\right)}{\sum_{v \in \mathcal{V}} \exp\left(f_{\text{con}}\ (v)^{\top} f_{\text{cen}}\ (v_{\text{cen}})\right)}$$

目标函数

共现信息

$$\mathcal{I} = \bigcup_{\mathcal{W} \in \mathcal{R}} \mathcal{I}_{\mathcal{W}}$$

尽可能预测正确

重构共现信息

利用节点嵌入预测出上下文节点

最大化

$$\prod_{\mathcal{W} \in \mathcal{R}} \prod_{(v_{cen}, v_{con}) \in \mathcal{I}_{\mathcal{W}}} \frac{\exp\left(f_{\text{con}}\left(v_{\text{con}}\right)^{\top} f_{\text{cen}}\left(v_{\text{cen}}\right)\right)}{\sum_{v \in \mathcal{V}} \exp\left(f_{\text{con}}(v)^{\top} f_{\text{cen}}\left(v_{\text{cen}}\right)\right)}$$

最小化

$$-\sum_{\mathcal{W} \in \mathcal{R}} \sum_{(v_{cen}, v_{con}) \in \mathcal{I}_{\mathcal{W}}} \log \frac{\exp\left(f_{\text{con}}\left(v_{\text{con}}\right)^{\top} f_{\text{cen}}\left(v_{\text{cen}}\right)\right)}{\sum_{v \in \mathcal{V}} \exp\left(f_{\text{con}}(v)^{\top} f_{\text{cen}}\left(v_{\text{cen}}\right)\right)}$$

提高可拓展性

$$p(v_{con}|v_{cen})$$

$$\frac{\exp\left(f_{\text{con}}\ (v_{\text{con}})^{\top} f_{\text{cen}}\ (v_{\text{cen}})\right)}{\sum_{v\in\mathcal{V}} \exp\left(f_{\text{con}}\ (v)^{\top} f_{\text{cen}}\ (v_{\text{cen}})\right)}$$

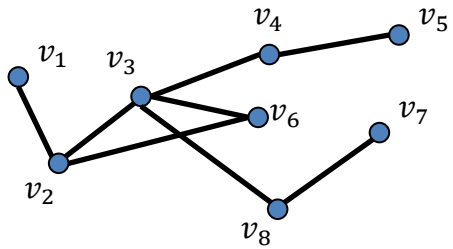
用其他方式来计算这个概率

Hierarchical Softmax

用其他方式来代替这一项

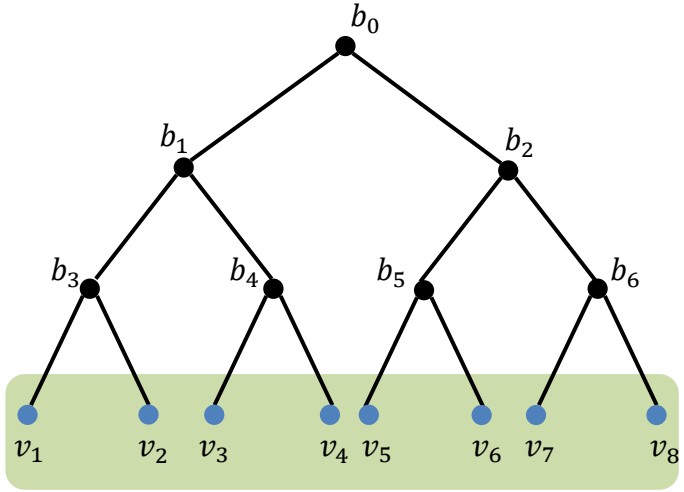
负采样

Hierchical Softmax



$$p(v_{con}|v_{cen})$$

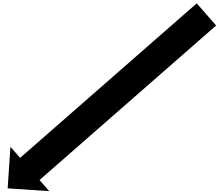
$$p\left(v_3 \mid v_8\right)$$



$$p\left(\text{left} \mid b_0, v_8\right) = \sigma\left(f_b\left(b_0\right)^\top f\left(v_8\right)\right)$$

负采样

最大化 $\dfrac{\exp\left(f_{\text{con}}\left(v_{\text{con}}\right)^{\top} f_{\text{cen}}\left(v_{\text{cen}}\right)\right)}{\sum_{v \in \mathcal{V}} \exp\left(f_{\text{con}}(v)^{\top} f_{\text{cen}}\left(v_{\text{cen}}\right)\right)}$ 最大化 / 最小化

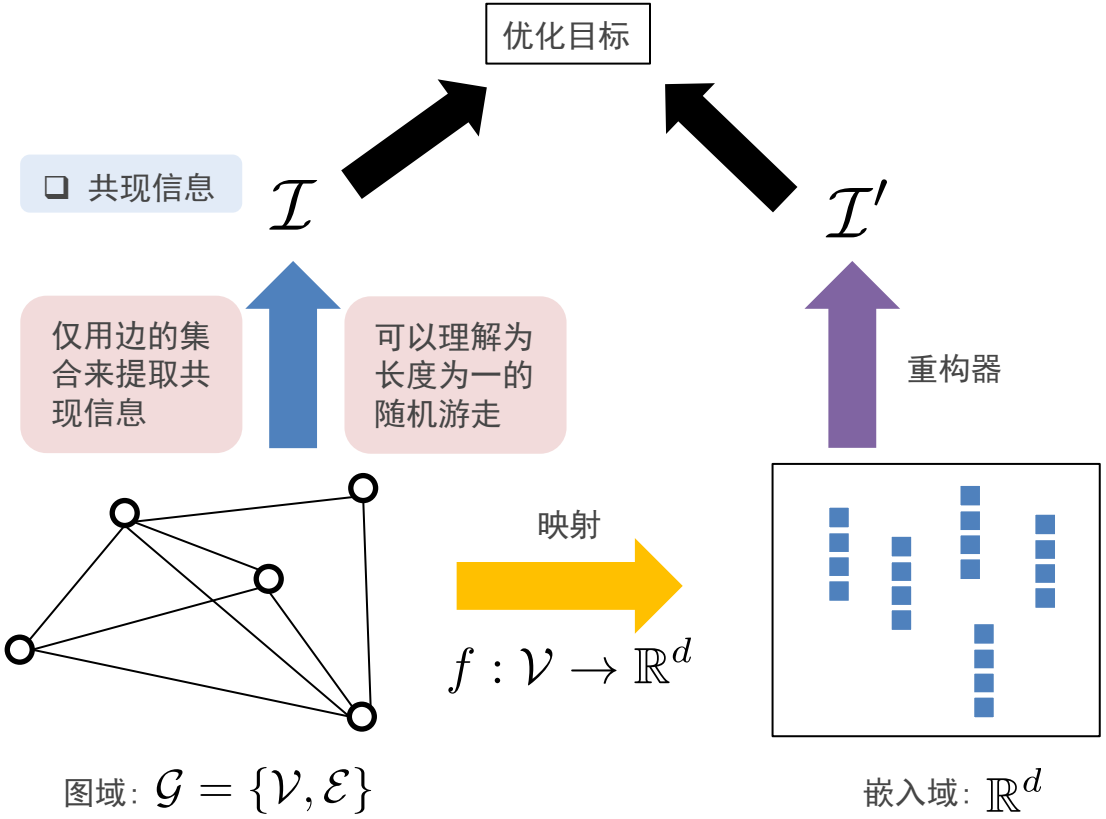


$$\log \sigma\left(f_{\text{con}}\left(v_{\text{con}}\right)^{\top} f_{\text{cen}}\left(v_{\text{cen}}\right)\right)+\sum_{i=1}^{k} E_{v_n \sim P_n(v)}\left[\log \sigma\left(-f_{\text{con}}\left(v_n\right)^{\top} f_{\text{cen}}\left(v_{\text{cen}}\right)\right)\right]$$

k个负样本

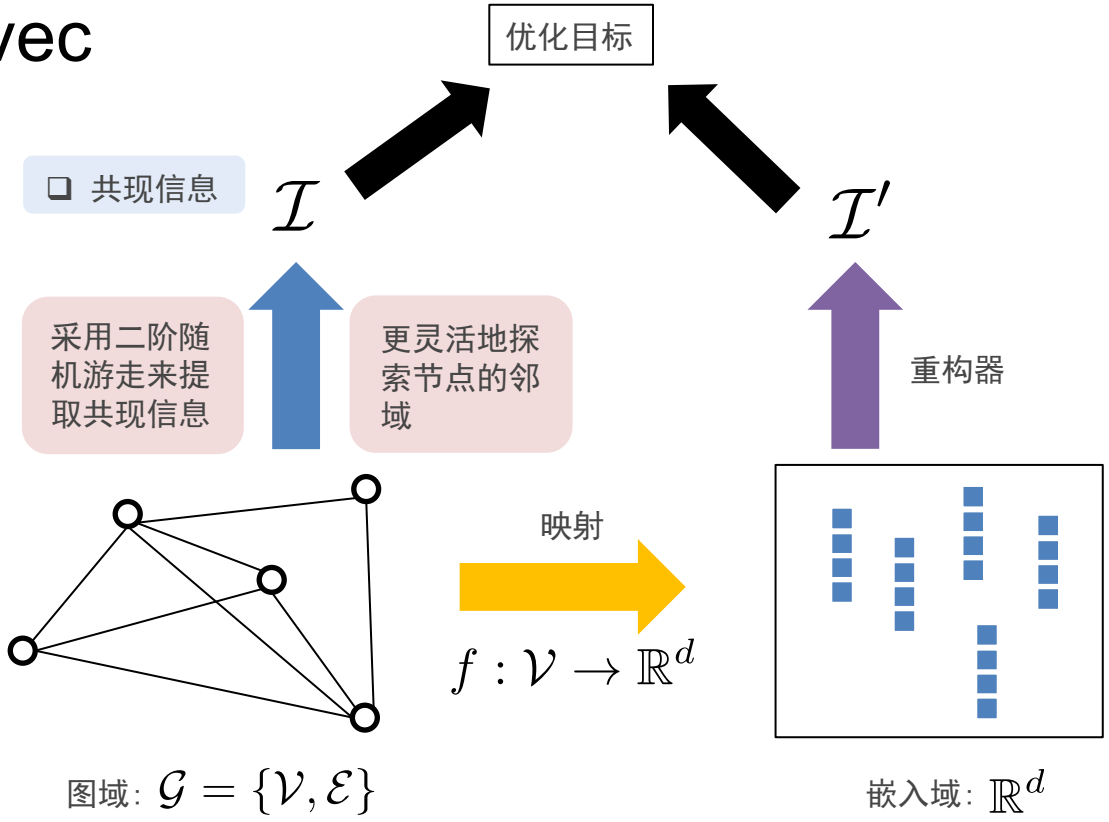
其他保留共现信息的图嵌入方法

LINE

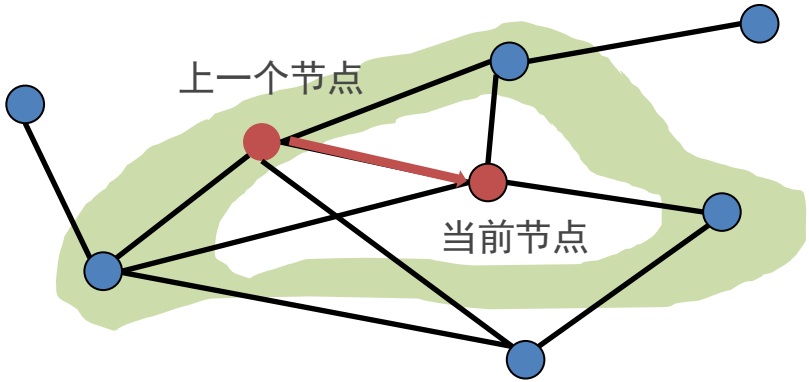


其他保留共现信息的图嵌入方法

node2vec



二阶随机游走



邻居节点

随机游走

随机选择一个邻居节点
作为下一个节点

二阶随机游走

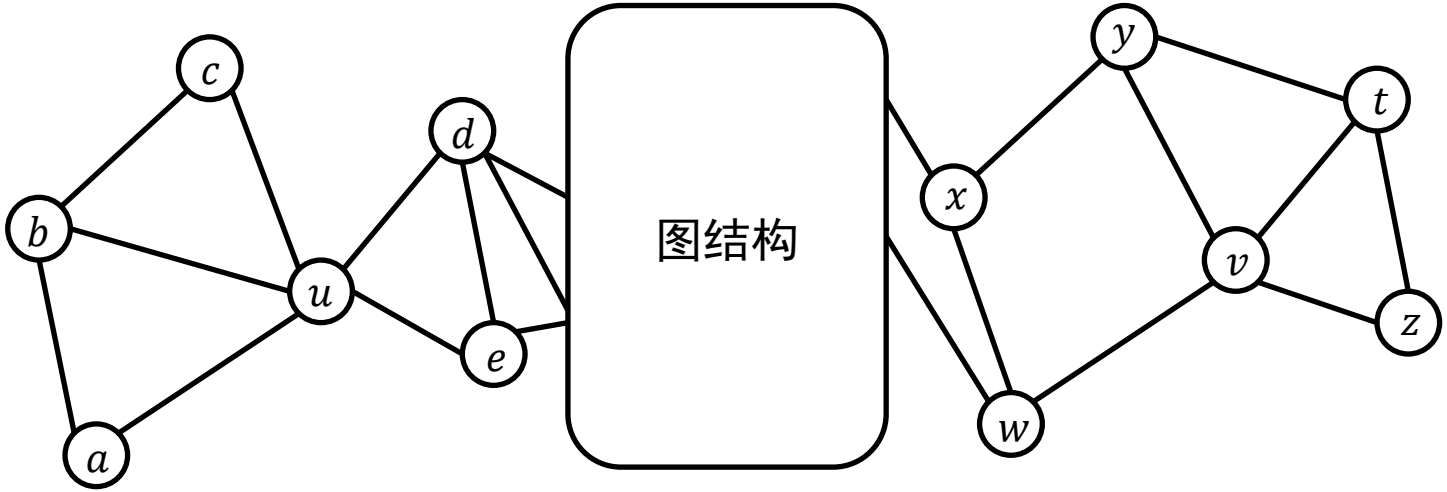
选择下一个节点时同时考虑
当前节点和上一节点

$$\alpha_{pq}\left(v^{(t+1)} \mid v^{(t-1)}, v^{(t)}\right) = \begin{cases} \frac{1}{p}, & \operatorname{dis}\left(v^{(t-1)}, v^{(t+1)}\right) = 0 \\ 1, & \operatorname{dis}\left(v^{(t-1)}, v^{(t+1)}\right) = 1 \\ \frac{1}{q}, & \operatorname{dis}\left(v^{(t-1)}, v^{(t+1)}\right) = 2 \end{cases}$$

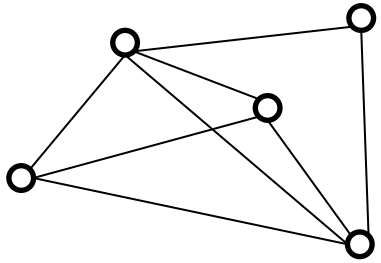
目录

- 简单图嵌入
  - 保留邻域信息
  - 保留结构角色
  - 保留节点状态
  - 保留社区结构

结构角色



保留结构角色信息的图嵌入:struc2vec



图

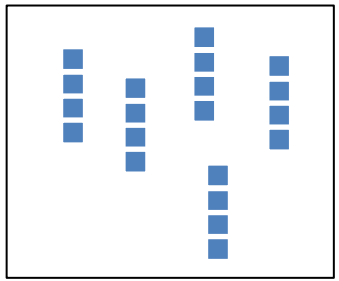
衡量角色结构相似性

角色结构相似性$g(v_i, v_j)$

基于角色结构相似性构建新图

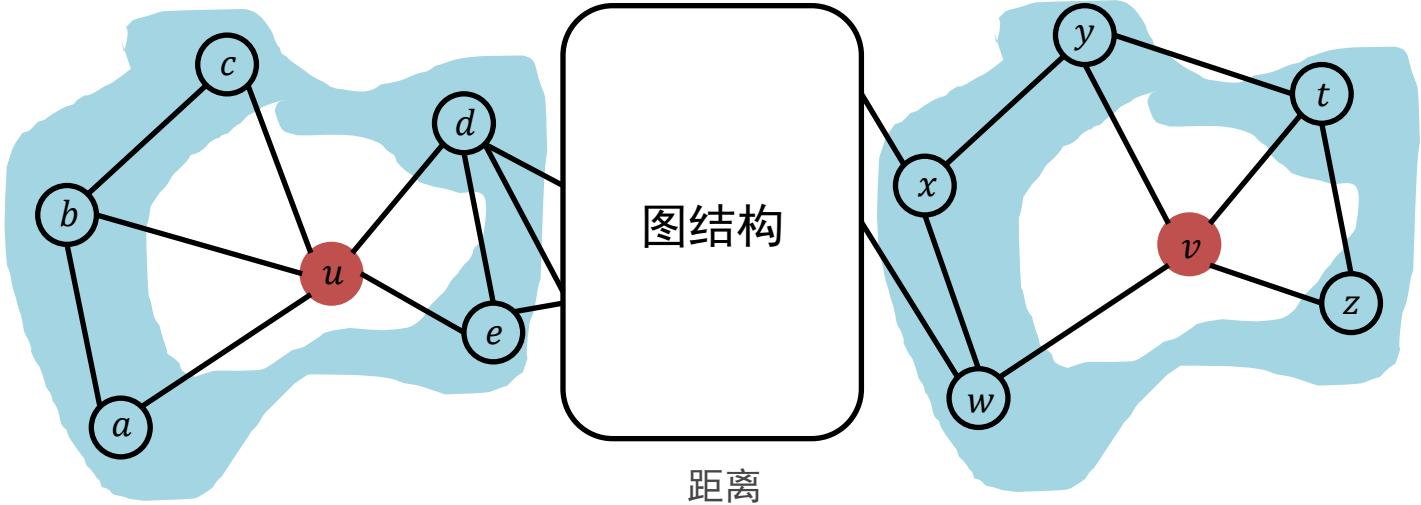
新图
（全连接）

DeepWalk



节点嵌入

衡量角色结构相似性



度相似

邻居的度相似

⋮

高阶邻居的度相似

距离

$dist(d_u, d_v)$

$+$

$dist(S_u^1, S_v^1)$

$+$

$dist(S_u^k, S_v^k)$

$=$

$g_k(u, v)$

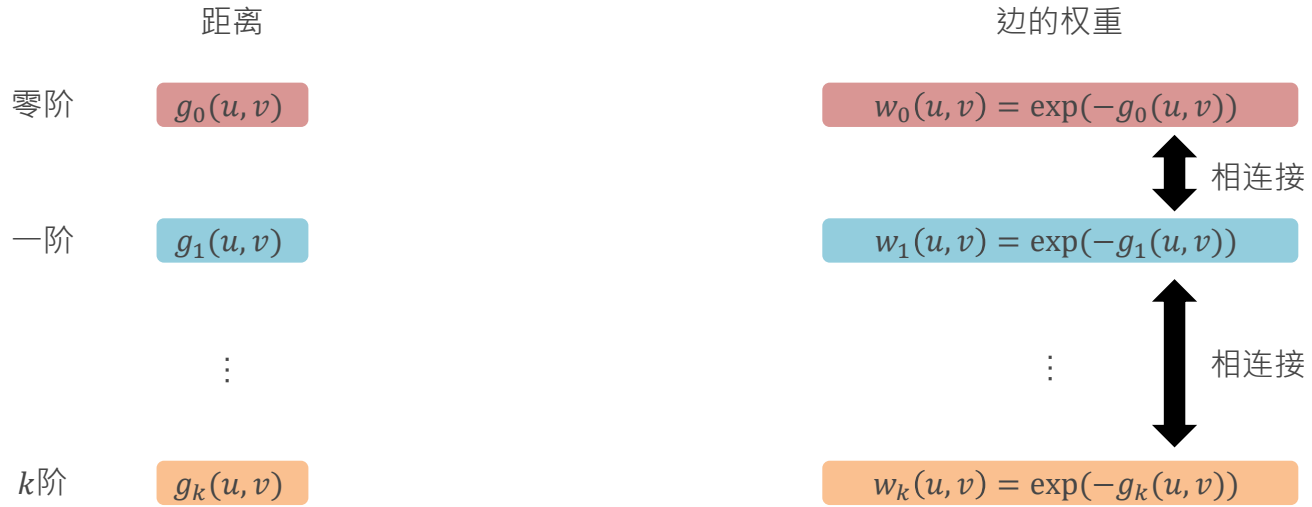
S_u^1：节点u的邻居的度的序列

根据度的大小排序

S_u^1：节点u的k阶邻居的度的序列

根据度的大小排序

全连接图的构造



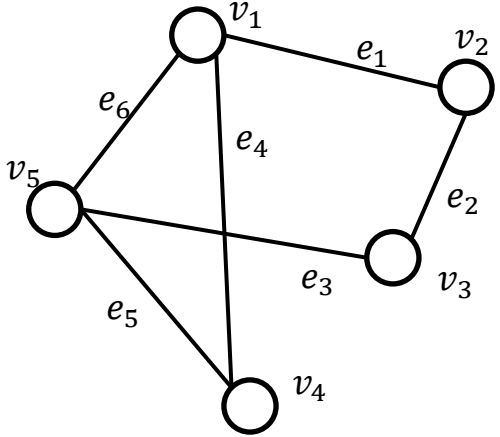
利用各阶的距离构造各自的全连接图

- 某阶内游走
- 相邻的阶之间游走

目录

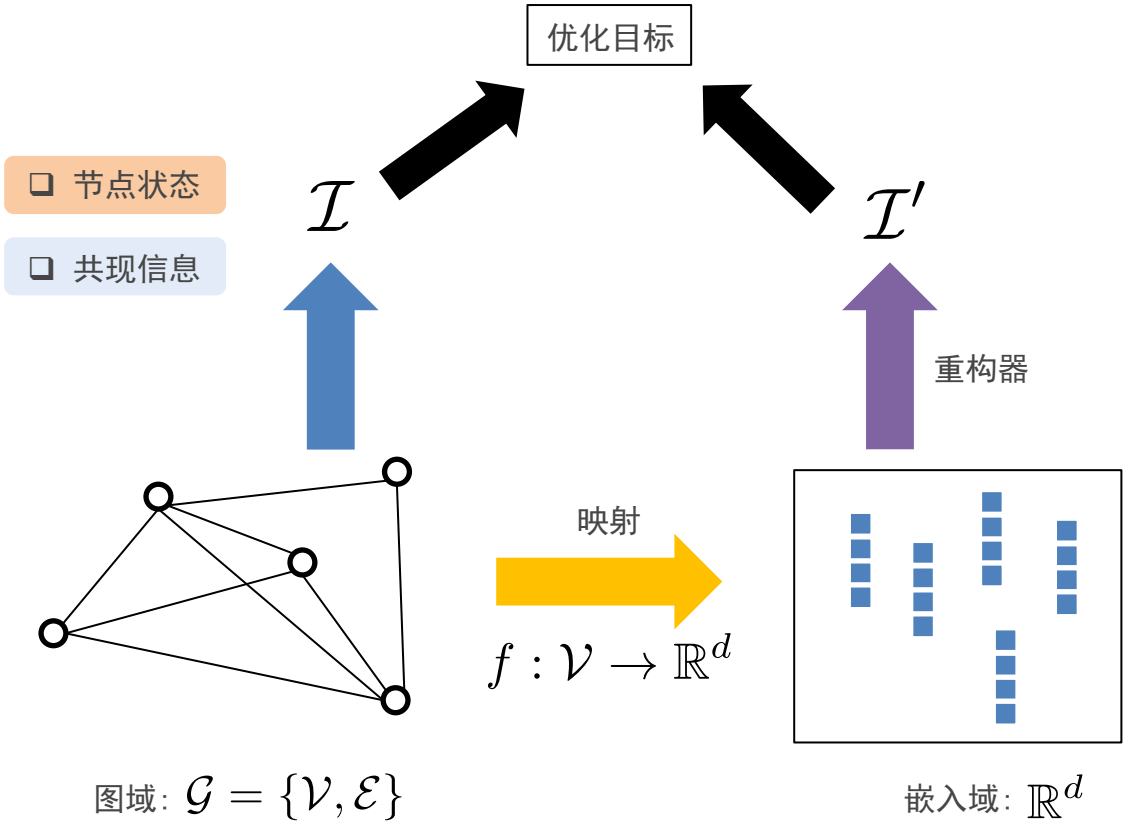
- 简单图嵌入
  - 保留邻域信息
  - 保留结构角色
  - 保留节点状态
  - 保留社区结构

节点状态

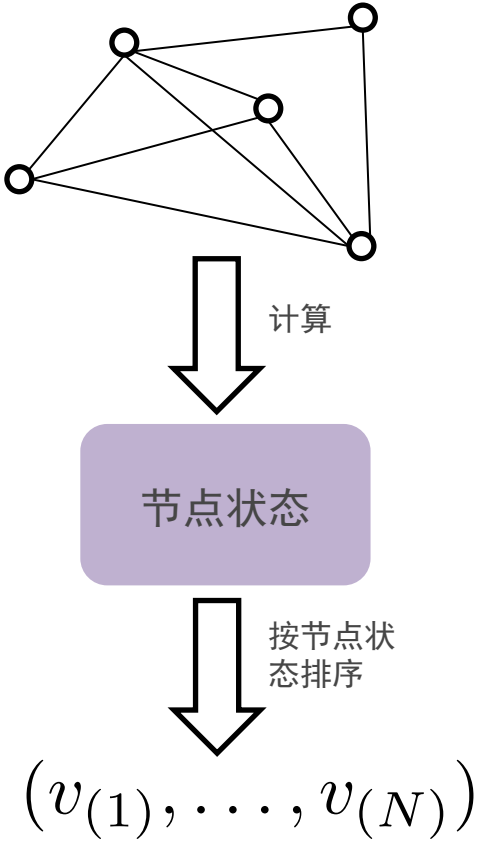


节点的中心性用来衡量
节点在图上的重要程度

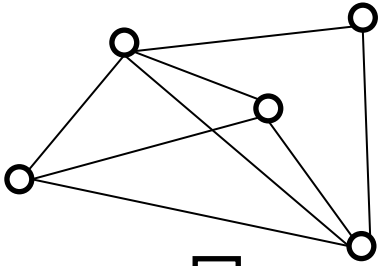
保留节点状态


优化目标
❑ 节点状态
❑ 共现信息
$\mathcal{I}$
$\mathcal{I}'$
重构器
映射
$f : \mathcal{V} \rightarrow \mathbb{R}^d$
图域: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$
嵌入域: $\mathbb{R}^d$

信息提取器



重构器



计算

节点状态

按节点状态排序

$$\left(v_{(1)}, \cdots, v_{(N)}\right)$$

重构节点状态信息

利用节点嵌入预测节点排序

$$p_{\text{global}} = \prod_{1 \leqslant i < j \leqslant N} p\left(v_{(i)}, v_{(j)}\right)$$

预测节点 $v_{(i)}$ 排在节点 $v_{(j)}$ 之前的概率

$$p\left(v_{(i),}, v_{(j)}\right) = \sigma\left(\boldsymbol{w}^{\top}\left(f(v_{(i)}) - f(v_{(j)})\right)\right)$$

目标函数

节点状态信息

$$(v_{(1)}, \ldots, v_{(N)})$$

尽可能预测正确

重构节点状态信息

利用节点嵌入预测节点排序

最大化

$$p_{\text{global}} = \prod_{1 \leqslant i < j \leqslant N} p\left(v_{(i)}, v_{(j)}\right)$$

实际中我们通常最小化它的对数的相反数

目录

- 简单图嵌入
  - 保留邻域信息
  - 保留结构角色
  - 保留节点状态
  - 保留社区结构

社区结构

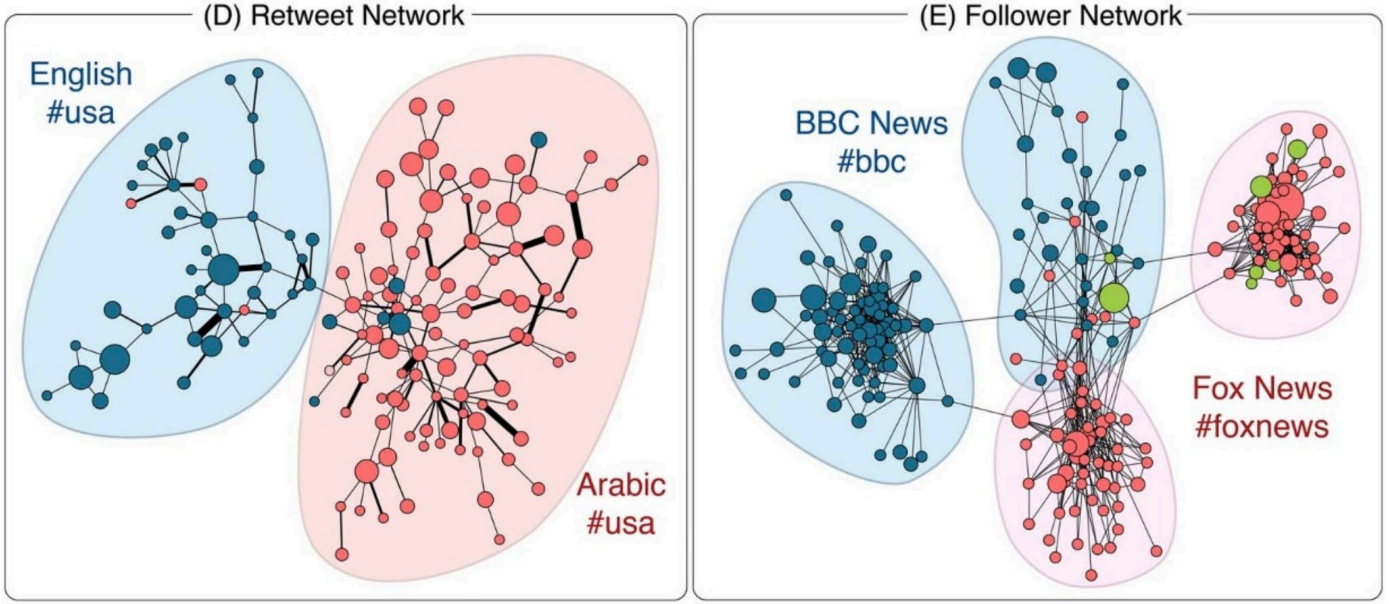


Image credit

- 社区内连接紧密
- 跨社区的连接不紧密

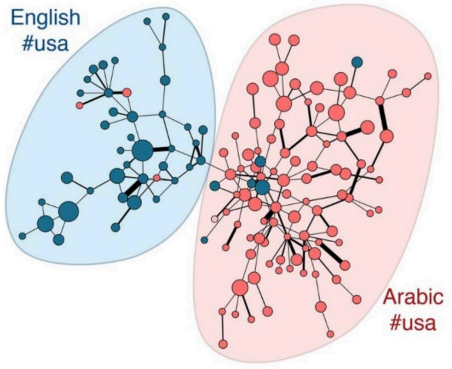
基于模块度的社区发现

模块度

$$Q = \frac{1}{2 \cdot \mathrm{vol}(\mathcal{G})} \sum_{ij} \left(\boldsymbol{A}_{i,j} - \frac{\mathrm{d}(v_i)\,\mathrm{d}(v_j)}{\mathrm{vol}(\mathcal{G})} \right) h_i h_j$$

$$Q = \frac{1}{2 \cdot \mathrm{vol}(\mathcal{G})} \boldsymbol{h}^\top \boldsymbol{B} \boldsymbol{h}$$

$$\boldsymbol{B}_{i,j}$$



拥有两个社区的图

$d(v_i)$ 节点的度

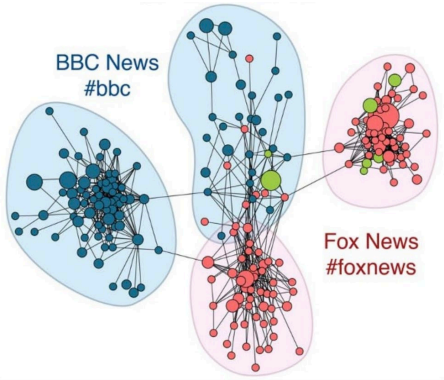
$vol(\mathcal{G})$ 所有节点的度的总和

h_i 节点所属社区的指示函数，取值0或1

$h_i = 1$ 或者 -1
表示节点v_i属于哪个社区

最大化模块度来进行社区发现

拓展到多个社区的情况



在图中有两个社区的情况下

$h_i = 1$ 或者 -1
表示节点v_i属于哪个社区

$\boldsymbol{h}$包含所有节点社区信息的向量

在图中有多于两个社区的情况下

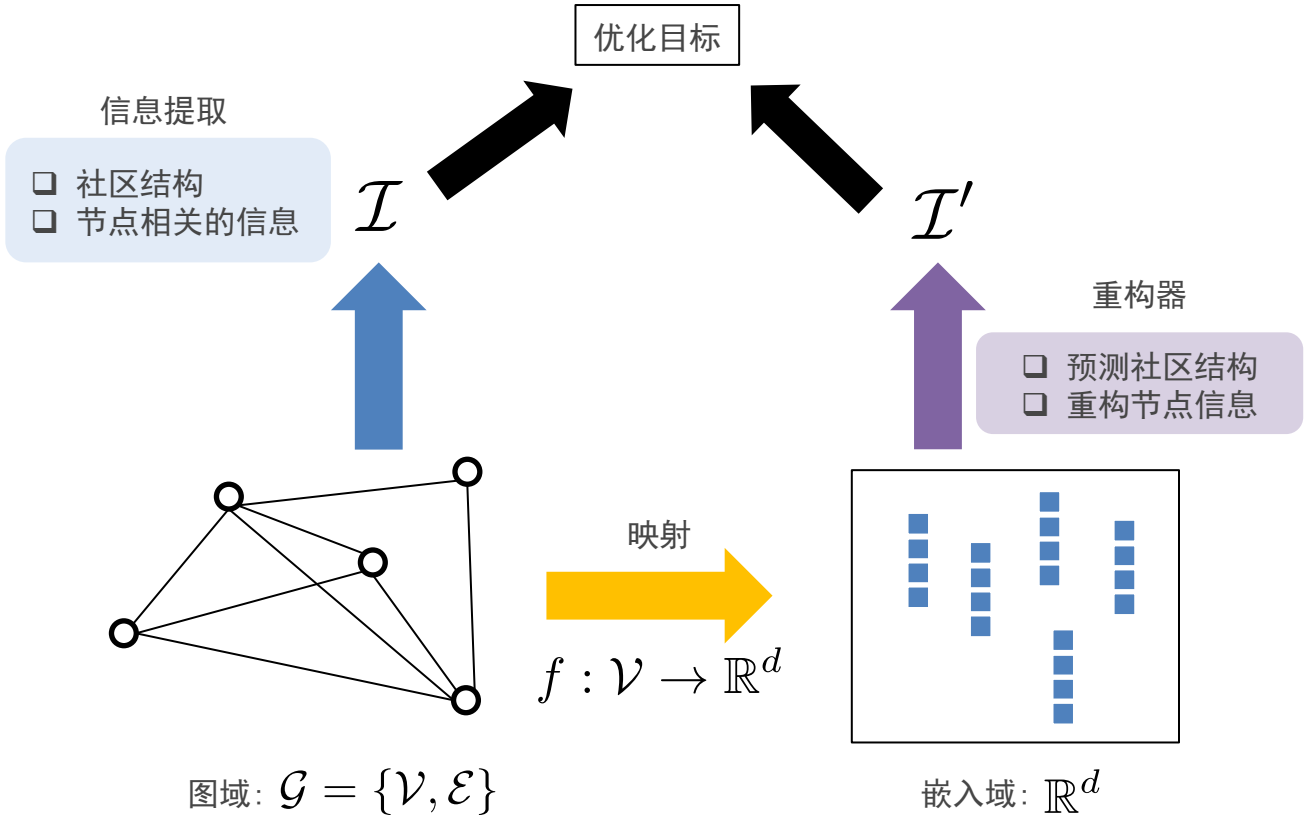
$\boldsymbol{H}$表示社区的分配矩阵

$\boldsymbol{H}$的每一行都是one-hot向量，表示了相应节点的社区归属

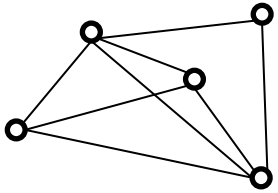
每个节点只属于一个社区

$$\text{s.t.} \quad \text{tr}\left(\boldsymbol{H}^\top \boldsymbol{H}\right) = N$$

保留社区结构的图嵌入


优化目标
信息提取
❑ 社区结构
❑ 节点相关的信息
$\mathcal{I}$
$\mathcal{I}'$
重构器
❑ 预测社区结构
❑ 重构节点信息
映射
$f : \mathcal{V} \rightarrow \mathbb{R}^d$
图域: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$
嵌入域: $\mathbb{R}^d$

提取节点相关的信息



连接矩阵
（边的信息）

A

邻域重合度信息

S

$$S_{i,j} = \frac{A_i A_j^\top}{\|A_i\| \, \|A_j\|}$$

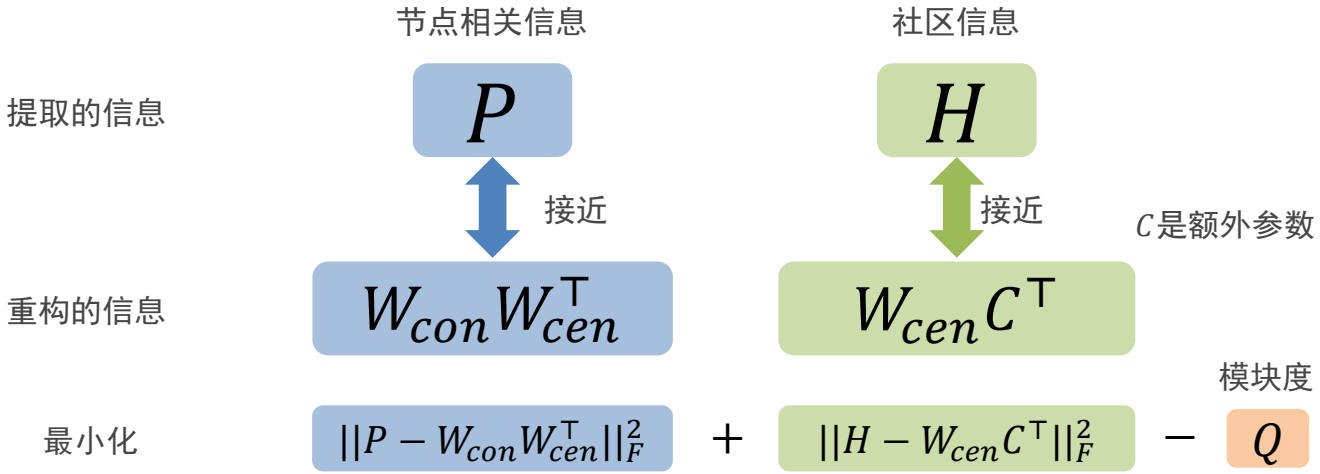
提取的节点信息 $P = A + \eta \cdot S$

重构器和目标函数

两个映射函数（和DeepWalk一样）

$$f_{cen}(v_i) = \mathbf{W}_{cen}\mathbf{e}_i \qquad f_{con}(v_i) = \mathbf{W}_{con}\mathbf{e}_i$$

节点嵌入



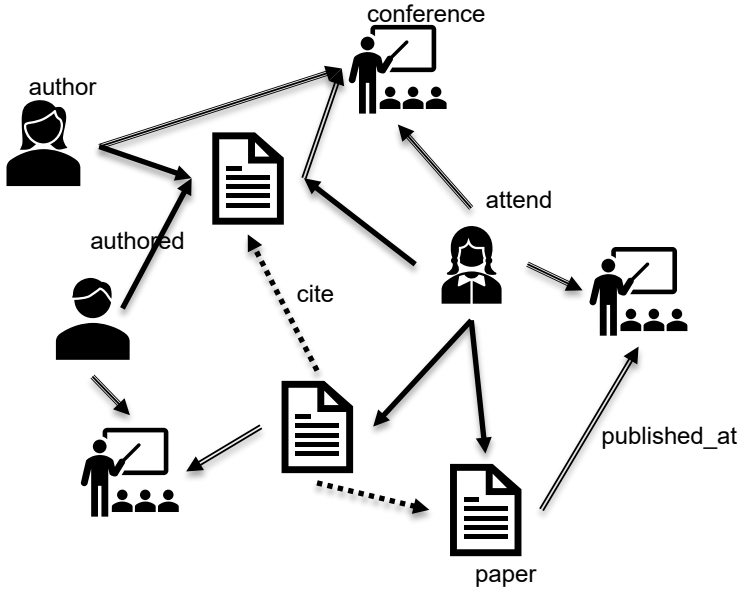
目录

目录

目录

- 复杂图嵌入
  - 异质图嵌入
  - 二分图嵌入
  - 多维图嵌入
  - 符号图嵌入
  - 超图嵌入
  - 动态图嵌入

异质图



$$\mathcal{V}, \mathcal{E}$$

包含不同种类的节点和边

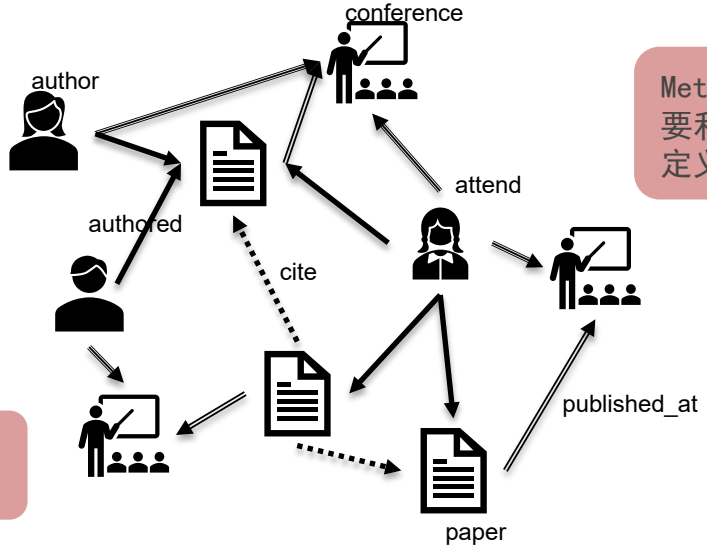
$$\phi_n : \mathcal{V} \to \mathcal{T}_n$$

$$\phi_e : \mathcal{V} \to \mathcal{T}_e$$

$$\mathcal{T}_n = \{\text{author}, \text{paper}, \text{conference}\}$$

$$\mathcal{T}_e = \{\text{authored}, \text{cite}, \text{published\_at}\}$$

Meta-path模式



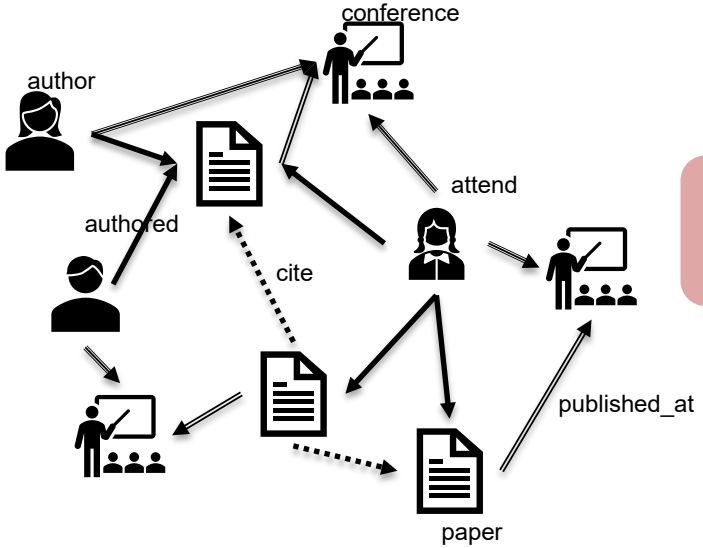
Meta-path模式一般需要利用专业知识来预先定义好的

不同的meta-path模式捕捉不同的语义信息

author → authored → paper ← authored ← author

author → authored → paper → cite → paper ← authored ← author

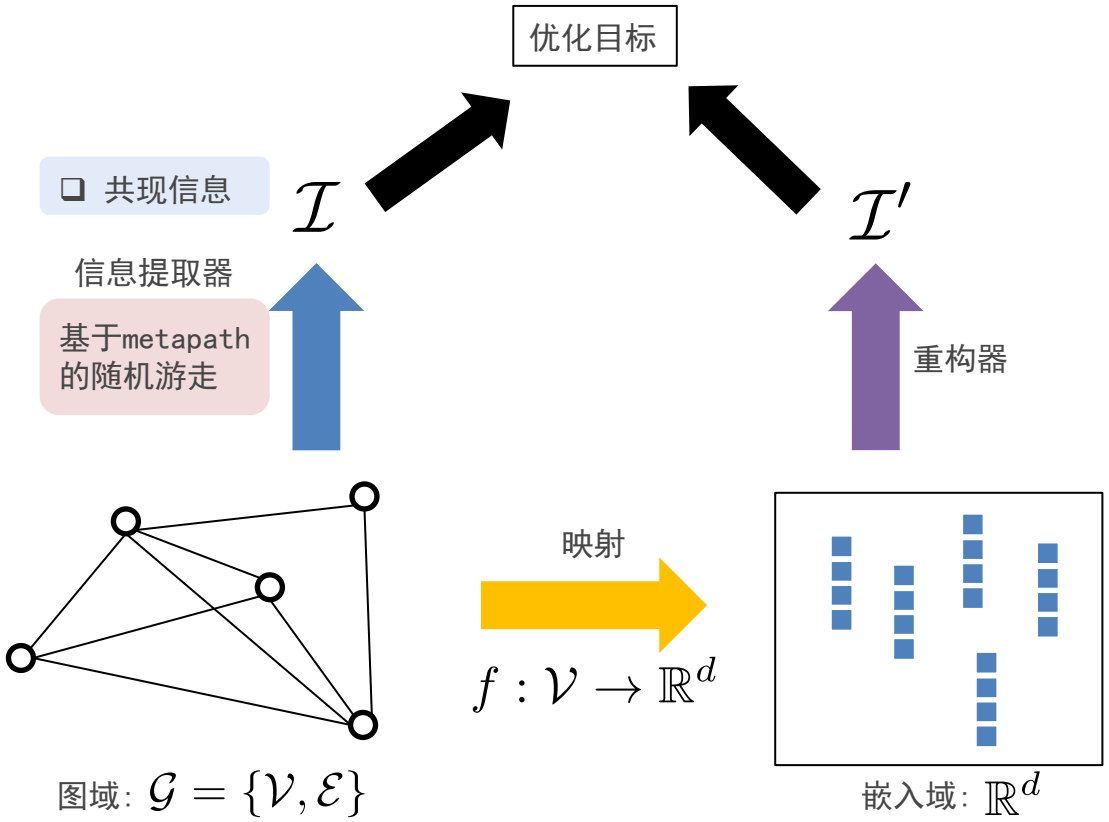
基于Meta-path的随机游走



一般选用多个预先定义好的meta-path模式来生成随机游走

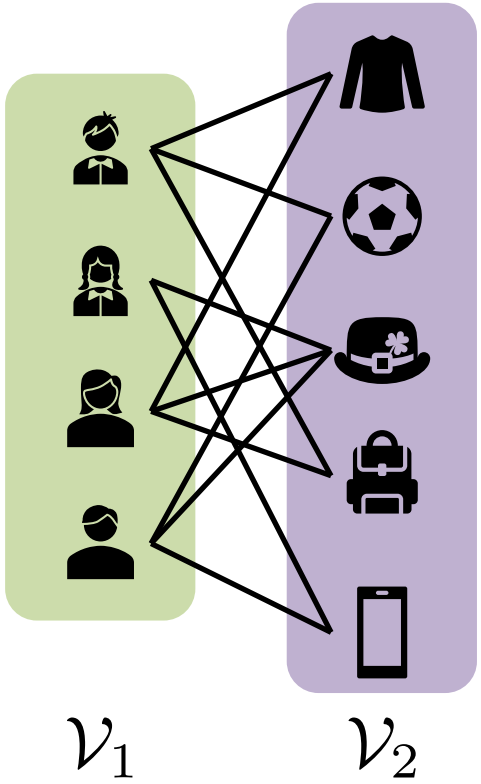
随机游走中的每个节点以及每条边都与meta-path模式中对应

异质图嵌入：metapath2vec



目录

二分图

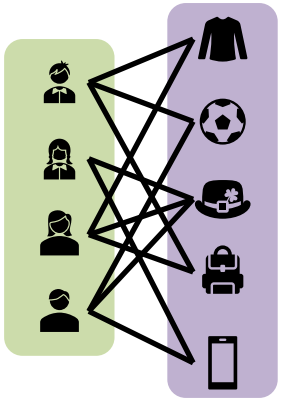


$$\mathcal{V}, \mathcal{E}$$

- $\mathcal{V}$可以被分为两个不相交的子集
- 每一子集内的任意点之间没有边相连接

信息提取器

- 如果 $\mathcal{V}_1$中的两个节点共享至少一个邻居，则这两个节点在生成子图中相连
- 新边的权重是两条原图中的边的乘积（如果共享多个邻居，则需要在这些邻居上求和）



$\mathcal{V}_1$ $\mathcal{V}_2$

$\mathcal{V}_1$生成的子图

$\mathcal{V}_1$中节点共现信息

Deepwalk

$\mathcal{E}$：边的信息

$\mathcal{V}_2$生成的子图

$\mathcal{V}_2$中节点共现信息

Deepwalk

边的信息的重构器和目标函数

$\mathcal{E}$ ：边的信息

重构边的信息

利用节点嵌入预测边是否存在

尽可能预测正确

$$e = (v^{(1)}, v^{(2)})$$

$$p(e|v^{(1)}, v^{(2)})$$

$$p(e|v^{(1)}, v^{(2)}) = \sigma\left(f(v^{(1)})^\top f(v^{(2)})\right)$$

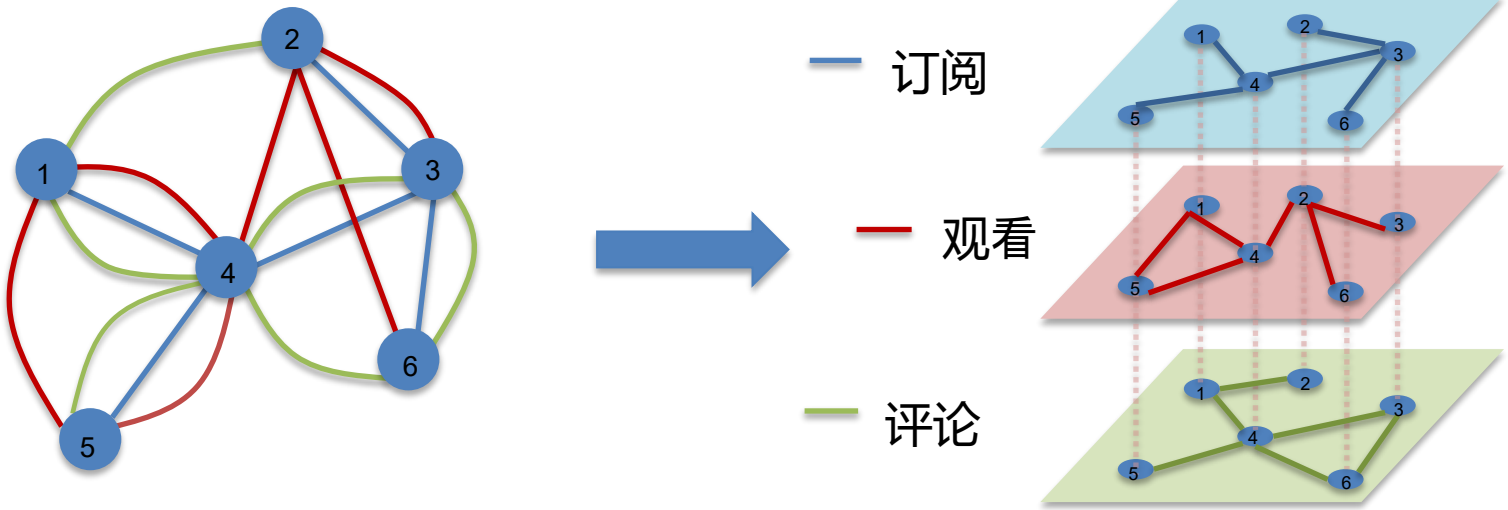
考虑所有边

最大化 $$\prod_{e\in\mathcal{E}} p(e|v^{(1)}, v^{(2)}) = \prod_{e\in\mathcal{E}} \sigma\left(f(v^{(1)})^\top f(v^{(2)})\right)$$

实际中我们通常最小化它的对数的相反数

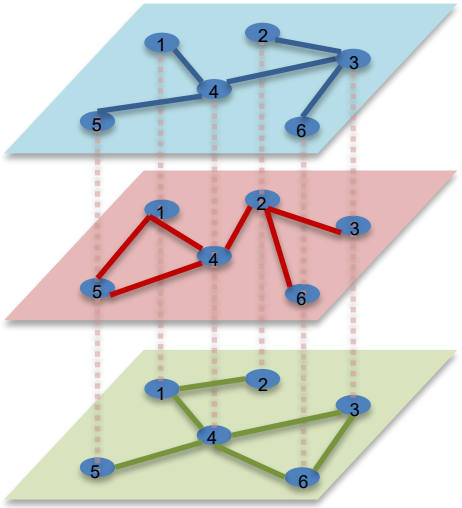
目录

多维图



同一对节点之间可以同时存在多种不同的关系

映射函数



通用映射

$f(v_i)$

维度特定映射

$f_{blue}(v_i)$

$f_{red}(v_i)$

$f_{green}(v_i)$

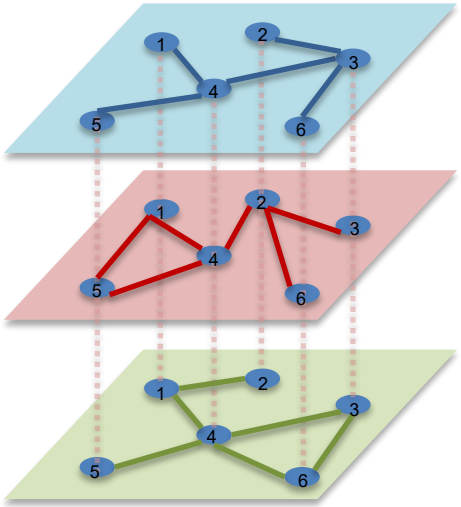
节点在某一维度的特定表示

$$\mathbf{u}_i^{blue} = f(v_i) + f_{blue}(v_i)$$

捕捉各个维度共享的信息

捕捉各个维度特有的信息

信息提取器



维度内随机游走

每个维度的共现信息

$\mathcal{I}_{blue}$

$\mathcal{I}_{red}$

$\mathcal{I}_{green}$

重构器和目标函数

每个维度的共现信息

利用节点在维度内的特定表示来重建每个维度的共现信息

$\mathcal{I}_{blue}$ ← $\{\mathbf{u}_i^{blue}\}_{i\in\mathcal{V}}$

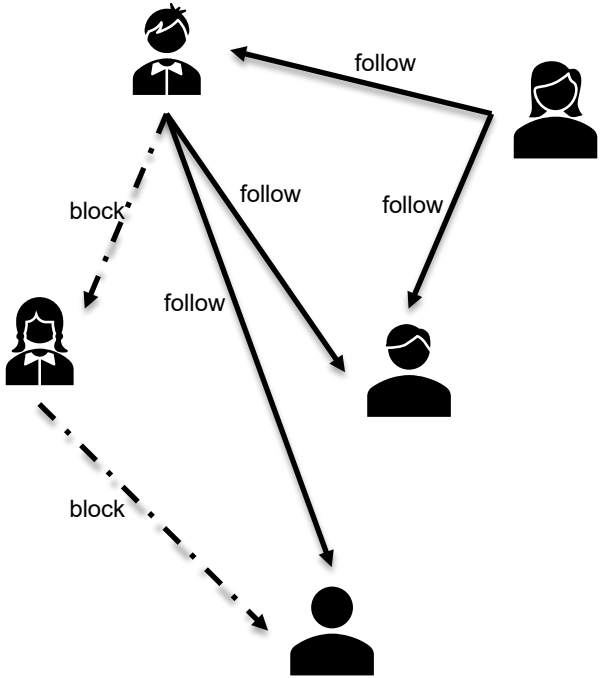
$\mathcal{I}_{red}$ ← $\{\mathbf{u}_i^{red}\}_{i\in\mathcal{V}}$

$\mathcal{I}_{green}$ ← $\{\mathbf{u}_i^{green}\}_{i\in\mathcal{V}}$

每一维度内的重建过程以及目标函数与deepwalk一致

目录

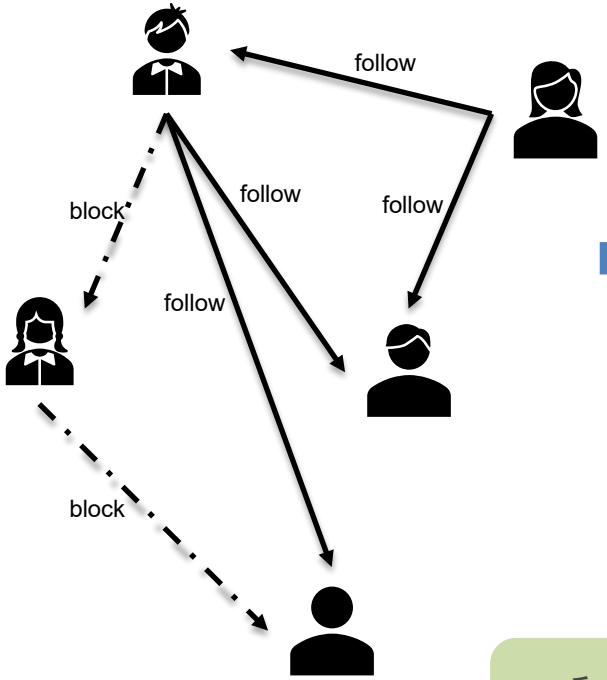
符号图

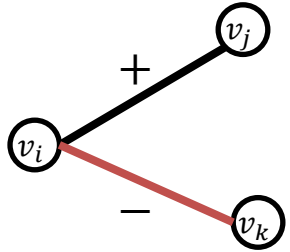


$$\mathcal{V}, \mathcal{E}$$

- $\mathcal{E}$可以被分为两个不相交的子集
- 这两个子集分别包含正边和负边

信息提取器





$\mathcal{I}$：所有这种形式的三元组

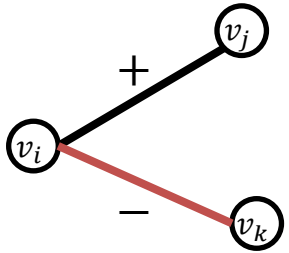
$$\mathcal{I} = \{(v_i, v_j, v_k) \mid \boldsymbol{A}_{i,j} = 1, \boldsymbol{A}_{i,k} = -1\}$$

v_i和 v_j的相似度高于
v_i和 v_k的相似度

重构器与目标函数

$$\mathcal{I} = \{(v_i, v_j, v_k) \mid \boldsymbol{A}_{i,j} = 1, \boldsymbol{A}_{i,k} = -1\}$$

v_i和v_j的相似度高于v_i和v_k的相似度



尽量好地重建

重建的相似度关系

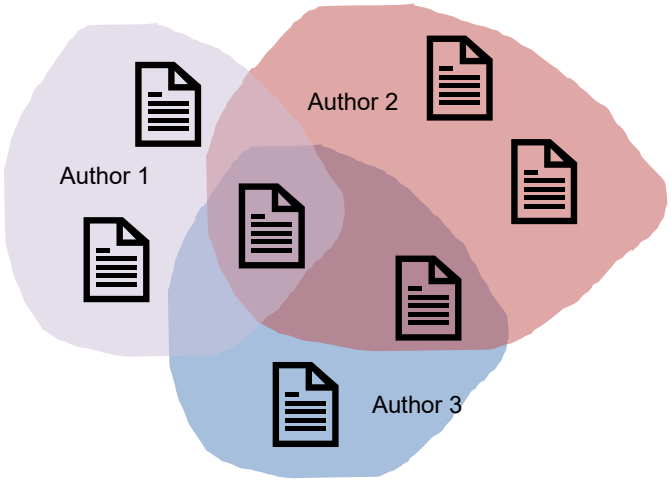
$$s\left(f\left(v_i\right), f\left(v_j\right)\right) - \left(s\left(f\left(v_i\right), f\left(v_k\right)\right) + \delta\right)$$

尽量使得它大于0

目录

- 复杂图嵌入
  - 异质图嵌入
  - 二分图嵌入
  - 多维图嵌入
  - 符号图嵌入
  - **超图嵌入**
  - 动态图嵌入

超图



$$\mathcal{V}, \mathcal{E}$$

- $\mathcal{E}$为超边的集合
- 超边可以描述超越两两之间的关系的更高维度的关系

信息提取器

$\mathcal{E}$：超边的信息

$\boldsymbol{A}$：节点在超边中的共现信息

$\boldsymbol{A}_{i,j}$:节点v_i和v_j在所有超边中共现的次数

$\boldsymbol{A}_i$： $\boldsymbol{A}$的第i行，表示节点v_i和其他所有节点的共现情况

重构器与目标函数：超边中的共现信息

映射函数

$$f(v_i) = MLP(A_i)$$

$\boldsymbol{A}$ ：节点在超边中的共现信息

尽可能接近

利用节点嵌入重构共现信息

$\boldsymbol{A_i}$：$\boldsymbol{A}$的第i行，表示节点v_i和其他所有节点的共现情况

$$\widehat{\boldsymbol{A}}_{\boldsymbol{i}} = MLP(f(v_i))$$

最小化 $$\sum_{v_i \in \mathcal{V}} \|\boldsymbol{A}_i - \hat{\boldsymbol{A}}_i\|_2^2$$

可以被考虑为自编码器

重构器与目标函数：超边

$\mathcal{E}$ ：超边的信息

重构超边的信息

尽可能预测正确

利用节点嵌入预测超边是否存在

$$e = \{v^{(1)}, v^{(2)}, v^{(3)}\}$$

$$p(e|\{v^{(1)}, v^{(2)}, v^{(3)}\})$$

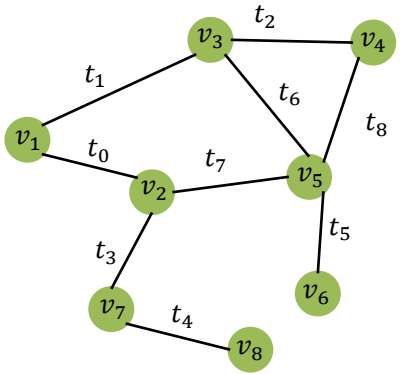
最大化 $$\prod_{e\in\mathcal{E}} p(e|\{v^{(1)}, v^{(2)}, v^{(3)}\}) = \prod_{e\in\mathcal{E}} \sigma\left(MLP([f(v^{(1)}), f(v^{(2)}), f(v^{(3)})])\right)$$

实际中我们通常最小化它的对数的相反数

目录

- 复杂图嵌入
  - 异质图嵌入
  - 二分图嵌入
  - 多维图嵌入
  - 符号图嵌入
  - 超图嵌入
  - 动态图嵌入

动态图



$$\mathcal{V}, \mathcal{E}$$

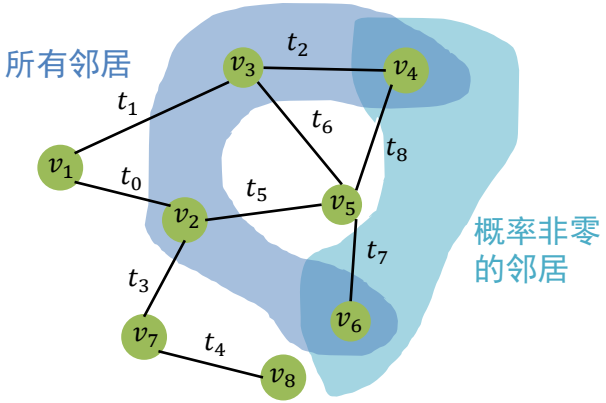
节点和边有对应的时间信息

$$\phi_n : \mathcal{V} \rightarrow \mathcal{T}$$

$$\phi_e : \mathcal{V} \rightarrow \mathcal{T}$$

$\mathcal{T}$表示时间的集合

时序随机游走



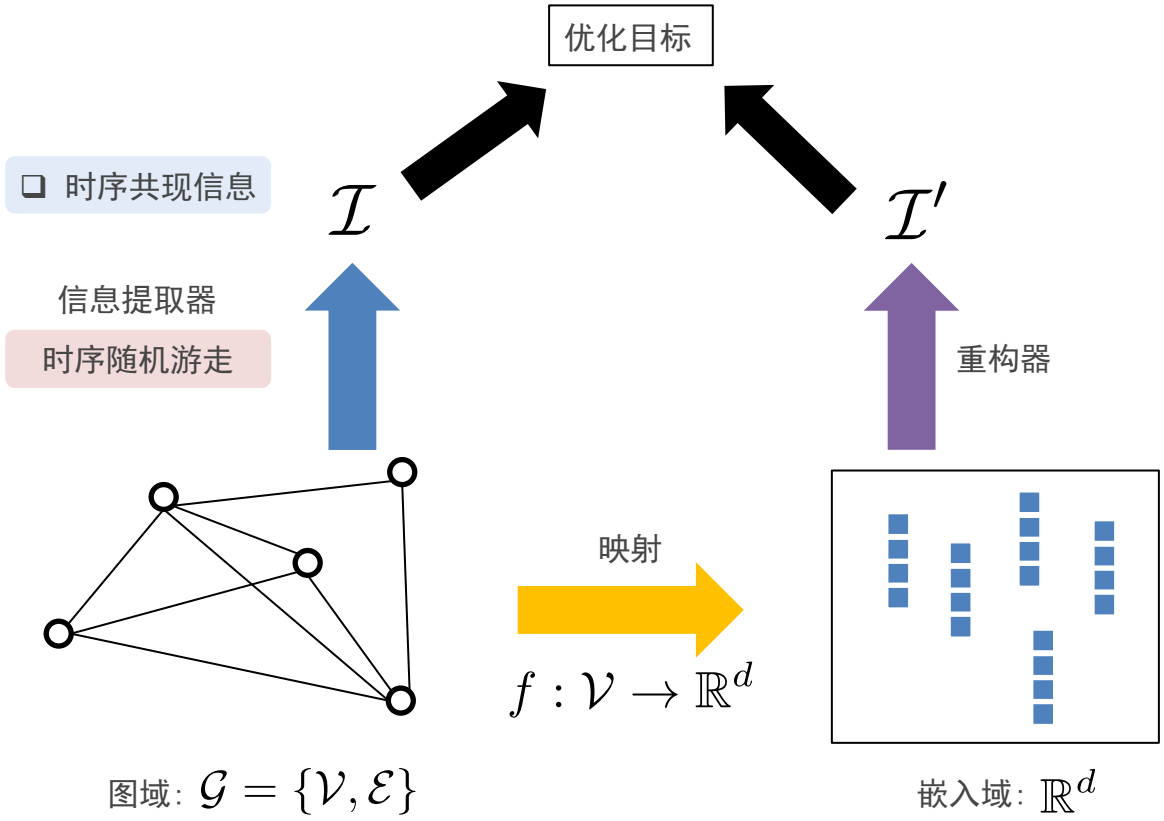
- 只能按时间顺序进行随机游走
- 下一条边的发生时间要在上一条边的发生时间之后

候选的下一节点　当前节点　　上一节点

$$p\left(v^{(k+1)} \mid v^{(k)}\right) = \begin{cases} \text{pre}\left(v^{(k+1)}\right), & \text{如果边}\left(v^{(k)}, v^{(k+1)}\right)\text{发生在}\left(v^{(k-1)}, v^{(k)}\right)\text{之后} \\ 0, & \text{如果边}\left(v^{(k)}, v^{(k+1)}\right)\text{发生在}\left(v^{(k-1)}, v^{(k)}\right)\text{之前} \end{cases}$$

会以较高的概率选择离当前时间具有较小间隔的节点

动态图嵌入


优化目标
❑ 时序共现信息
$\mathcal{I}$
$\mathcal{I}'$
信息提取器
时序随机游走
重构器
映射
$f : \mathcal{V} \rightarrow \mathbb{R}^d$
图域: $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$
嵌入域: $\mathbb{R}^d$

目录

深蓝学院
shenlanxueyuan.com

感谢聆听！

Thanks for Listening