

图神经网络导论

深度社区发现

授课教师：周晟

浙江大学 软件学院

2022.12



课程内容

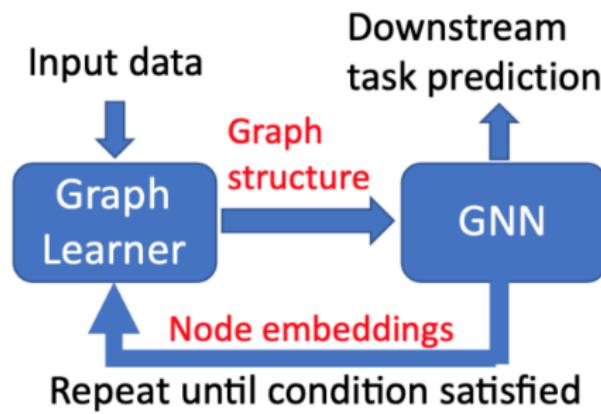
- ① 研究背景
- ② 经典社区发现算法
- ③ 基于自训练的社区发现算法
- ④ 基于迭代优化的社区发现算法
- ⑤ 基于对比学习的社区发现
- ⑥ 社区发现的端到端应用



上节课回顾

图结构学习

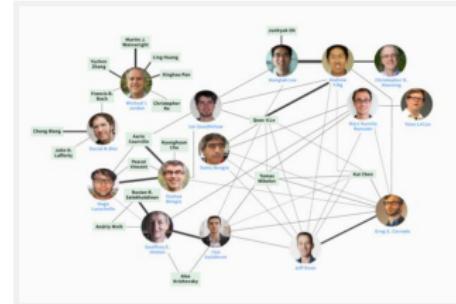
图结构与图神经网络学习可以相互帮助：好的图结构可以学习好的图神经网络，好的图神经网络结果可以帮助建模更准确的网络结构。



生活中的社区



社交网络



引用网络



交通网络



疫情传播

社区结构

物以类聚，人以群分

社区结构局部成因：

- ① 具有相似的属性/标签的节点容易产生交互
- ② 具有不同属性/标签的节点容易产生分离

社区结构全局成因：

- ① 连接关系有限
- ② 对立关系存在



社区的特点

关系强弱

社区内用户的关系往往更为紧密，而社区间的用户关系往往较为稀疏。

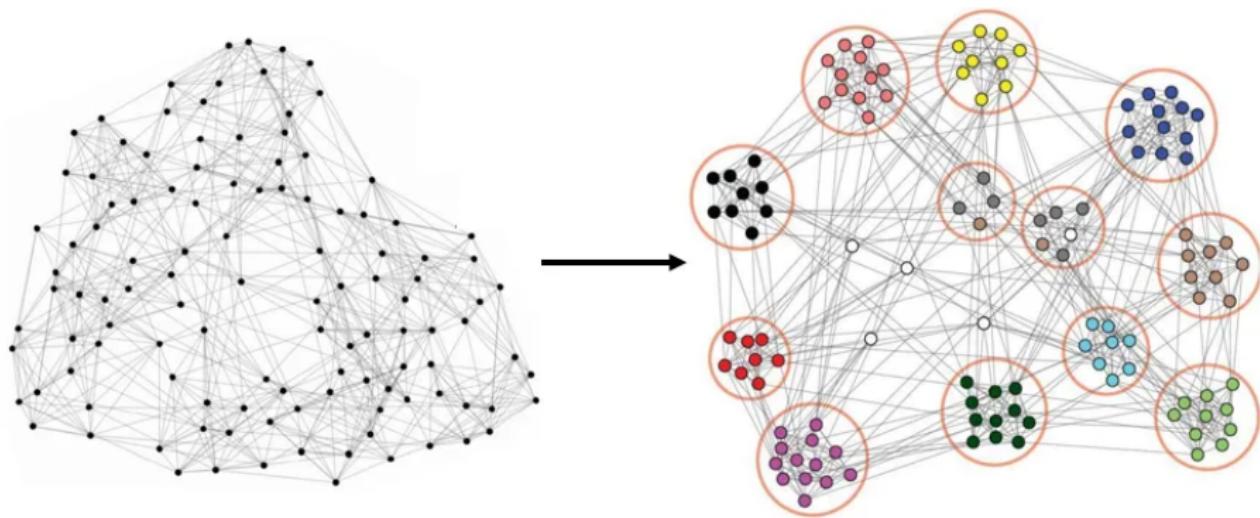
信息量

社区内的信息往往较为单一且容易冗余，社区间往往能传递有价值的信息。

社区结构与消息传递机制：

社区内连边往往满足同质性假设 (Homophily)，社区间连边往往是异质的 (Heterophily)

社区和社区发现



社区发现的定义

社区

社区 (Community) 是指图上具有相似特性且稠密连接的子图。

社区发现

社区发现 (Community Detection) 也称为图聚类 (Graph/Network Clustering)，目标是自动发现网络中具有相同标签或稠密连接的子图。

社区发现与图聚类的区别？

社区发现的定义

社区

社区 (Community) 是指图上具有相似特性且稠密连接的子图。

社区发现

社区发现 (Community Detection) 也称为图聚类 (Graph/Network Clustering)，目标是自动发现网络中具有相同标签或稠密连接的子图。

社区发现与图聚类的区别？

社区内的节点，一般连接比较稠密，而不只是属性或标签相似。

社区发现的意义

社交网络：

- ① 疾病传播控制
- ② 定向广告投放

电商网络：

- ① 风险社团识别
- ② 用户兴趣建模
- ③ 社交推荐系统



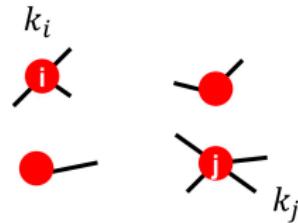
- ① 研究背景
- ② 经典社区发现算法
- ③ 基于自训练的社区发现算法
- ④ 基于迭代优化的社区发现算法
- ⑤ 基于对比学习的社区发现
- ⑥ 社区发现的端到端应用



Modularity

- 给定一个有 n 个节点和 m 条边的图 G , 已知每个节点 i 的度数 k_i
- 边是均匀随机连接的
- 两个节点之间可以存在多条边

节点 i 和节点 j 之间的期望边数为:



$$k_i \cdot \frac{k_j}{2m} = \frac{k_i k_j}{2m}$$

- 有向边的总数为 $2m$
- 从节点 i 出发的边有 k_i 条, 以节点 j 为终点的概率为 $\frac{k_j}{2m}$,
 $\frac{1}{2m}$ 能够将 Q 标准化到 $[-1, 1]$ 范围内

Modularity

给定社区的定义（社区内连接稠密，社区间连接稀疏），定义度量社区质量的指标 Modularity：

Modularity Q

给定一种将网络 G 划分为若干社区的方式 $s \in S$ ，Modularity Q 定义为：

$$Q(G, S) = \frac{1}{2m} \sum_{s \in S} \sum_{i \in s} \sum_{j \in s} \left(A_{ij} - \frac{k_i k_j}{2m} \right)$$

$$Q \propto \sum_{s \in S} [(\text{观测的社区 } s \text{ 内部边数}) - (\text{随机的社区 } s \text{ 内部边数})]$$

Modularity

进一步表示

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j) = \frac{1}{2m} \sum_c \left[\sum_{in} in - \frac{(\sum_{tot})^2}{2m} \right]$$

- A_{ij} 表示节点 i 和 j 之间的边权
- k_i 和 k_j 分别为连接到节点 i 和节点 j 的边的权重之和
- $2m$ 为图中所有边的权重之和
- c_i 和 c_j 为节点所属的社区
- δ 为指示函数 $\delta(c_i, c_j) = 1$ if $c_i = c_j$ else 0
- \sum_{in} 表示社区 C 内的边权和
- \sum_{tot} 表示连接到社区 C 内的边权和



Fast Unfolding of Communities in Large Networks (Louvain)

研究动机

给定社区质量的评价指标 Modularity，Louvain 算法通过贪心的策略寻找使得 Modularity 最大的社区划分。

Louvain 算法使用自底向上，逐层聚类的迭代策略，基本步骤：

- ① 将每个节点加入能够使 modularity 增益最大的邻居社区中
- ② 将社区归纳为新的节点

迭代上述操作直至收敛。



Louvain 算法的核心步骤：

- ① 初始化：将每个节点看作一个独立的社区
- ② 社区重分配：
 - 计算将每个节点 v_i 分配到其邻居节点 $v_j, i \in \mathcal{N}_i$ 所在的社区，计算模块度增益 ΔQ_{ij}
 - 将节点转移到模块度增益最大的那个社区
- ③ 图重构：将重分配之后的社区作为新的社区节点，社区内的边权和为社区节点自环边的权重，社区节点间的边权为社区间边权的和。
- ④ 迭代社区重分配和图重构步骤，直至收敛



Modularity 变化

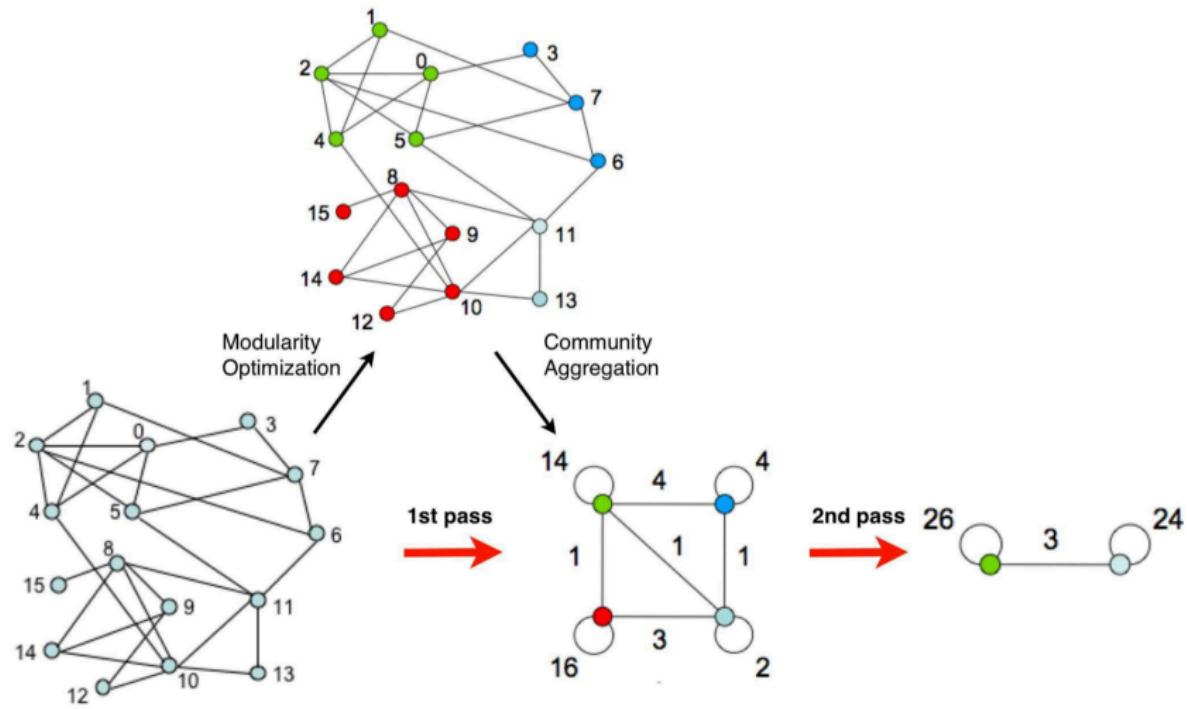
将节点 v_i 分配至社区 C 带来的 Modularity 增益

$$\Delta Q = \left[\frac{\sum_{in} + k_{i,in}}{2m} - \left(\frac{\sum_{tot} + k_i}{2m} \right)^2 \right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m} \right)^2 - \left(\frac{k_i}{2m} \right)^2 \right]$$

- \sum_{in} 表示社区 C 内的边权和
- \sum_{tot} 表示连接到社区 C 内的边权和
- k_i 为节点 v_i 相连的边权和
- $k_{i,in}$ 为社区 C 的节点与节点 v_i 的边权和
- m 是全图的边权和



Louvain



优点

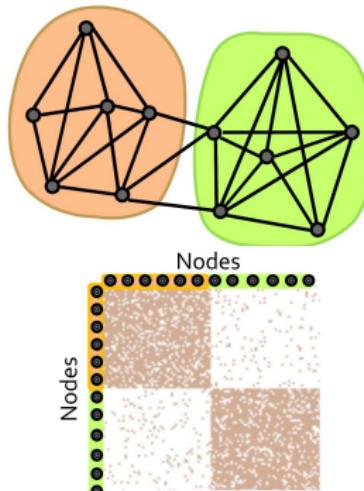
- ① 优化目标明确
- ② 优化效率高
- ③ 解释性强

优点

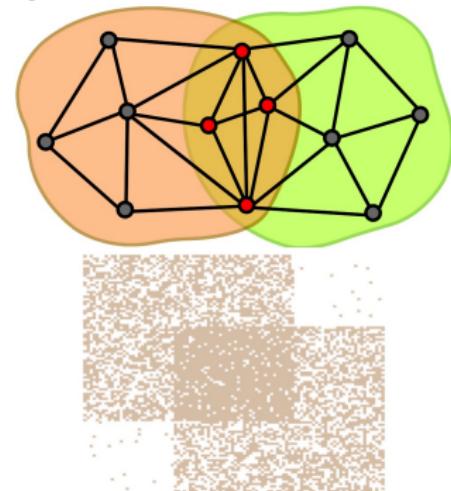
- ① 容易受到异常点的影响
- ② 容易受到较大边权的边的影响
- ③ 容易受到稀疏性的影响
- ④ 无法融合复杂信息
- ⑤ 无法解决**重叠社区**问题

Overlapping community detection at scale: a nonnegative matrix factorization approach (BigCLAM)

BigCLAM 是一种用于在网络中挖掘重叠社区的方法。它基于节点社区隶属关系，通过生成模型来发现社区。



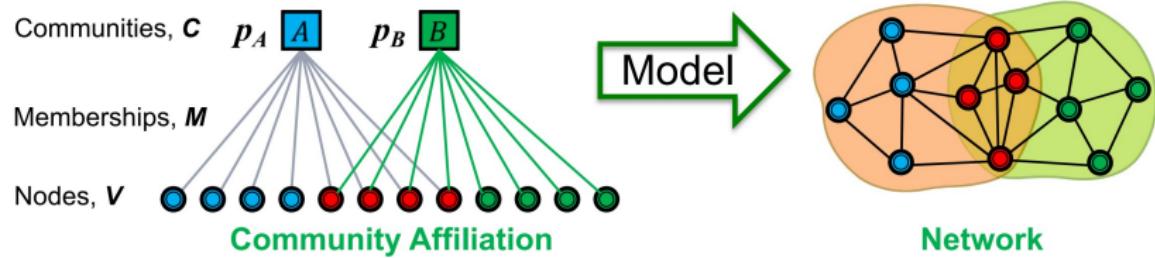
非重叠社区



重叠社区

社区关系图模型 AGM

如何从社区关系中生成一个网络¹？



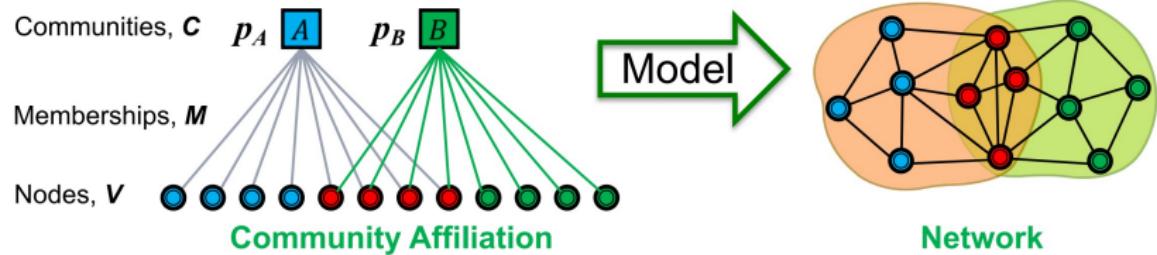
模型参数：

- 节点集 V , 社区集 C , 从属关系 M
- 每个社区 c 有一个对应的概率 p_c

¹Overlapping community detection at scale: a nonnegative matrix factorization approach(WSDM 2013)



社区关系图模型 AGM



给定参数 $(V, C, M, \{p_c\})$:

- 社区 c 中的节点之间有 p_c 的概率相互连接
- 属于多个社区的节点有多次独立的采样机会，由此节点 u 和 v 之间存在边的概率为：

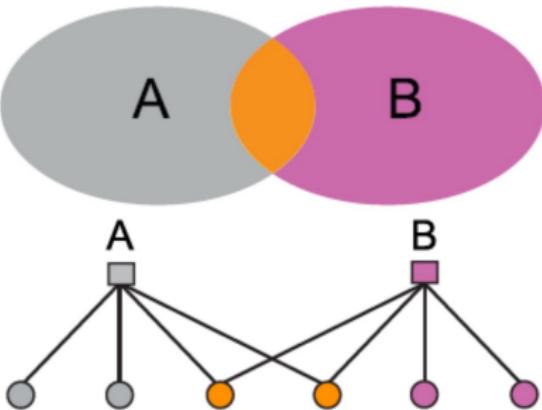
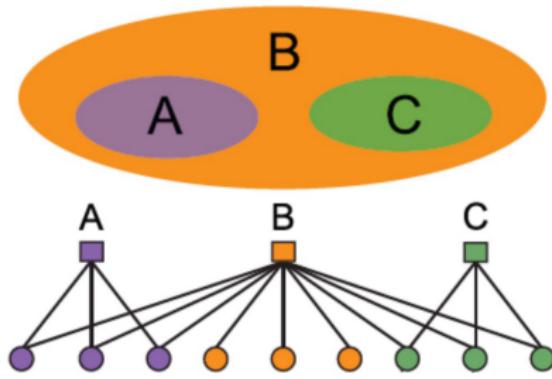
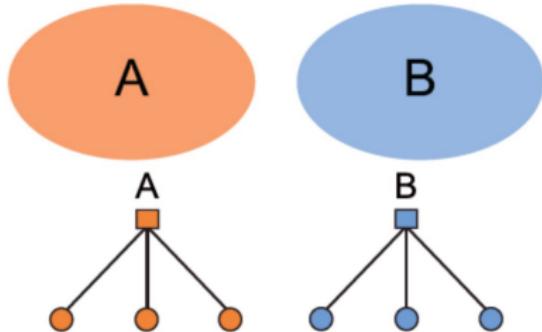
$$p(u, v) = 1 - \prod_{c \in M_u \cap M_v} (1 - p_c)$$

- 注意：若节点 u 和 v 没有共同社区，则 $p(u, v) = 0$



AGM 的灵活性

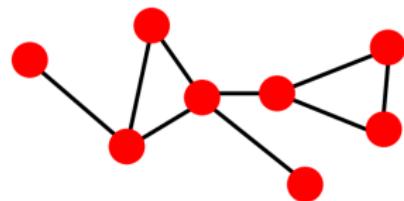
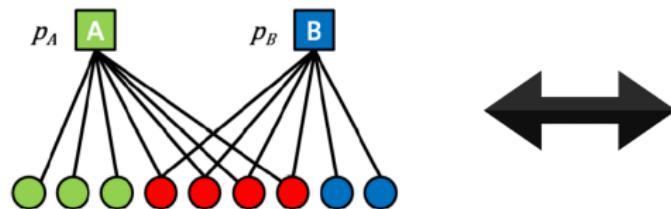
- AGM 能够表达各种社区的结构：
 - 嵌套
 - 无重叠
 - 重叠



使用 AGM 发现社区

给定一个图 G , 我们需要找到模型 F , 具体就是:

- 节点和社区的从属关系 M
- 社区数量 C
- 概率参数 p_c



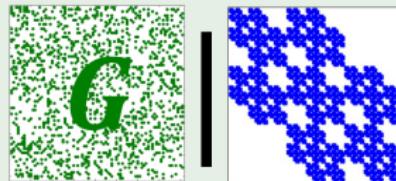
对于图 G , 如何估计模型参数?



图似然

最大似然估计

- 给定一个真实图 G , 我们想找到模型参数 F 使得:

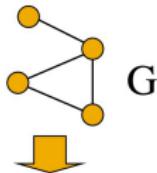
$$\arg \max_F P(G | F)$$


- 为了解决这个问题, 我们需要
 - 高效地计算 $P(G|F)$
 - 学习 F 参数使其最大化



计算 $P(G|F)$

- 给定 G 和 F , 我们计算由 F 生成 G 的似然: $P(G|F)$



F

0.25	0.10	0.10	0.04
0.05	0.15	0.02	0.06
0.05	0.02	0.15	0.06
0.01	0.03	0.03	0.09

$P(u, v)$: 边 (u, v) 存在的概率

1	0	1	1
0	1	0	1
1	0	1	1
1	1	1	1

$P(G|F)$

G

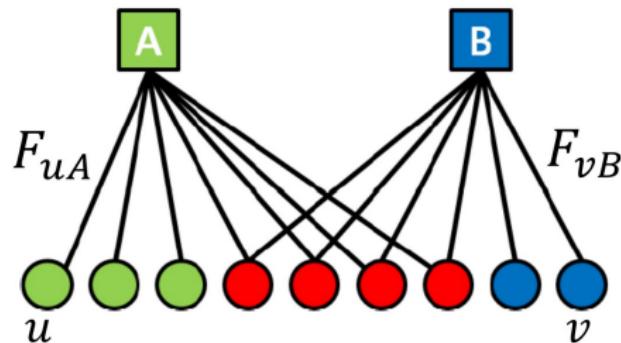
$$P(G|F) = \prod_{(u,v) \in G} P(u, v) \prod_{(u,v) \notin G} (1 - P(u, v))$$

图中存在边的似然

图中不存在边的似然

更一般的 AGM

在更一般的 AGM 中，节点与社区之间的隶属关系是有概率/权重的



- F_{uA} : 节点 u 属于社区 A 的概率/权重
- $F_{uA} = 0$ 表示没有任何隶属关系



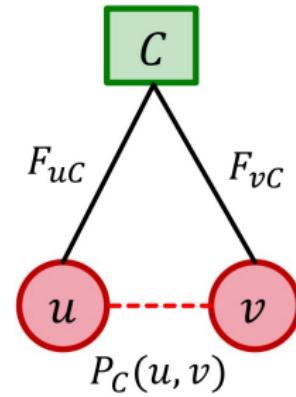
更一般的 AGM

$$P_C(u, v)$$

对于社区 C , 我们可以将节点 i 和节点 j 之间存在边的概率写为:

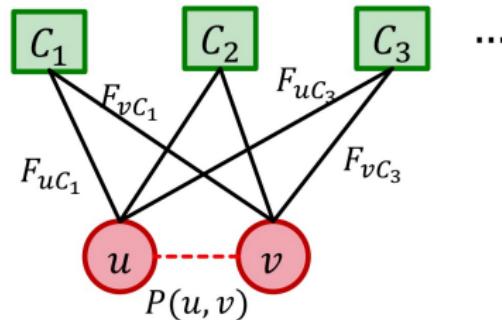
$$P_C(u, v) = 1 - \exp(-F_{uC} \cdot F_{vC})$$

- $F_{uC} \cdot F_{vC} \geq 0$, 因此 $P_C(u, v)$ 满足
 $0 \leq P_C(u, v) \leq 1$
- $P_C(u, v) = 0$ if $F_{uC} \cdot F_{vC} = 0$
- $P_C(u, v) \approx 1$ if $F_{uC} \cdot F_{vC}$ is large



更一般的 AGM

- 当然，节点 u 和 v 也有可能通过多个共享的社区来连接



- 节点 u 和 v 至少通过一个社区来连接的概率为：

$$P(u, c) = 1 - \prod_{C \in \Gamma} (1 - P_C(u, c))$$

Γ 为所有社区的集合。



BigCLAM 模型

节点连接概率

节点 u, v 之间连接的概率与其共享的关系权重正相关：

$$P(u, v) = 1 - \exp(-F_u^T F_v)$$

- 给定图 $G(V, E)$, 我们想最大化 G 在当前模型下的似然:

$$\begin{aligned} P(G|F) &= \prod_{(u,c) \in E} P(u, v) \prod_{(u,c) \notin E} (1 - P(u, v)) \\ &= \prod_{(u,c) \in E} (1 - \exp(-F_u^T F_v)) \prod_{(u,c) \notin E} \exp(-F_u^T F_v) \end{aligned}$$

BigCLAM 模型

数值不稳定

似然涉及到许多小概率的连乘，可能会导致数值的不稳定。

因此，我们使用对数似然来代替：

$$\begin{aligned} & \log(P(G|F)) \\ &= \log \left(\prod_{(u,c) \in E} (1 - \exp(-F_u^T F_v)) \prod_{(u,c) \notin E} \exp(-F_u^T F_v) \right) \\ &= \sum_{(u,v) \in E} \log(1 - \exp(-F_u^T F_v)) - \sum_{(u,v) \notin E} F_u^T F_v \\ &\equiv \mathcal{L}(F) : \text{即目标函数} \end{aligned}$$



目标函数

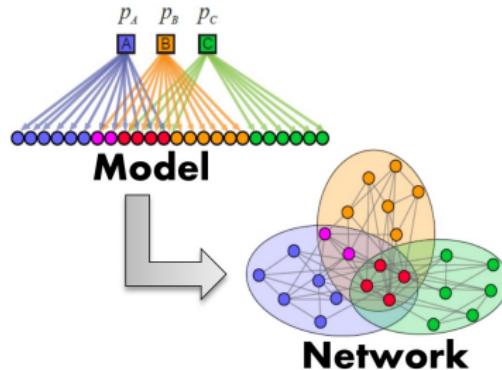
$$\mathcal{L}(F) = \sum_{(u,v) \in E} \log(1 - \exp(-F_u^T F_v)) - \sum_{(u,v) \notin E} F_u^T F_v$$

- 训练从**随机初始化**的参数开始，逐步迭代到收敛
- 对于节点 $u \in V$ ，**固定其他节点的社区关系权重**，只更新它自己的社区关系权重 F_u



BigCLAM 总结

- BigCLAM 定义模型来生成具有重叠社区结构的网络。



- 给定一个图, BigCLAM的参数 (每个节点的社区关系权重)
可以通过最大化模型生成图的对数似然来估计。

Community Detection in Networks with Node Attributes (CESNA)

模型假设²:

- ① 属于相同/相似社区的节点容易产生边
- ② 一个节点可以同时属于多个社区
- ③ 同时属于多个社区的节点比只属于同一个社区的节点有更大的概率生成边
- ④ 属于同一个社区的节点有更大概率有相似的属性



²Community Detection in Networks with Node Attributes (ICDM 2013)

网络中的边建模：

$$P_{uv}(c) = 1 - \exp(-F_{uc} \cdot F_{vc})$$

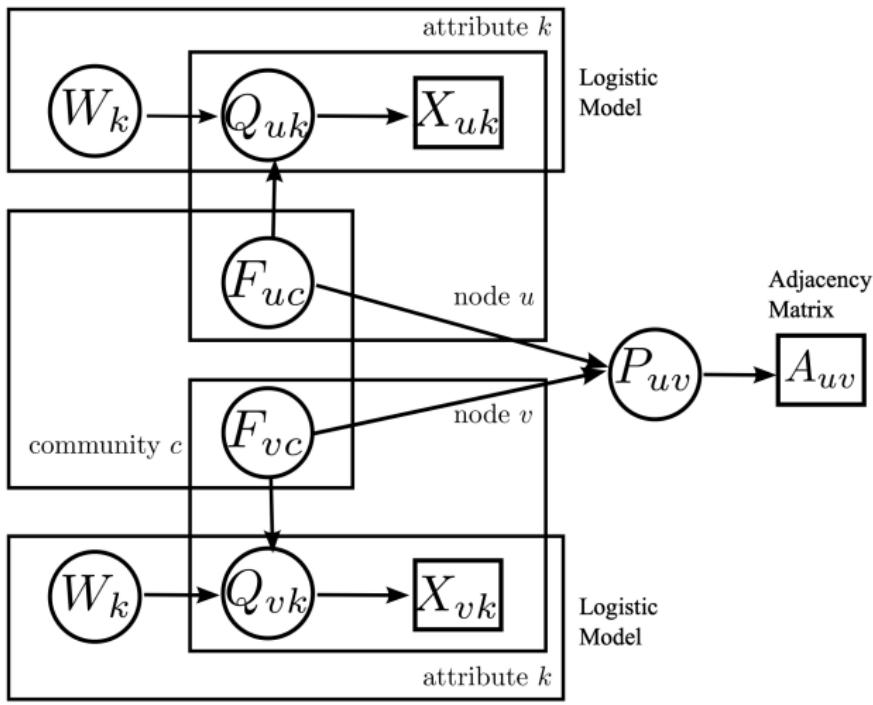
其中 F_{vc} 是节点 v 属于社区 c 的概率。节点间观测到边的概率可以定义为：

$$\begin{aligned} P_{uv} &= 1 - \exp\left(-\sum_c F_{uc} \cdot F_{vc}\right), \\ A_{uv} &\sim \text{Bernoulli}(P_{uv}). \end{aligned}$$

CESNA 假设，节点的属性是由所属的社区生成的。

$$\begin{aligned} Q_{uk} &= \frac{1}{1 + \exp\left(-\sum_c W_{kc} \cdot F_{uc}\right)} \\ X_{uk} &\sim \text{Bernoulli}(Q_{uk}) \end{aligned}$$





经典社区发现算法

经典社区发现算法的主要挑战：

- ① 社区的假设与标签不一定对齐
- ② 难以融合节点或边的特征
- ③ 难以建模节点间的复杂关系

深度学习 + 社区发现 = 深度社区发现



- ① 研究背景
- ② 经典社区发现算法
- ③ 基于自训练的社区发现算法
- ④ 基于迭代优化的社区发现算法
- ⑤ 基于对比学习的社区发现
- ⑥ 社区发现的端到端应用



基于自训练的社区发现算法

社区发现的核心困难：

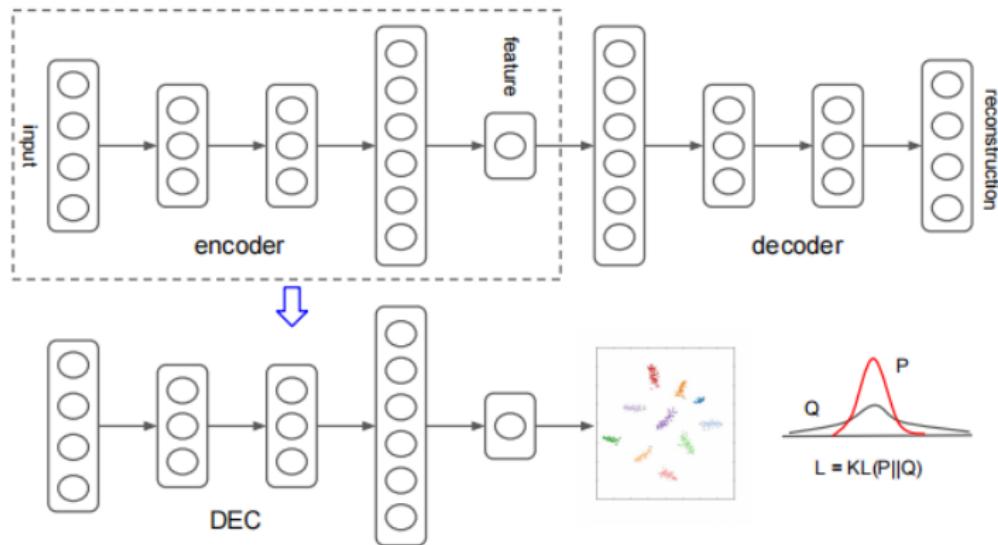
- ① 无监督
- ② 节点间关系复杂
- ③ 多目标优化

基于自训练的社区发现算法

- ① 不依赖外部监督信号
- ② 使用图神经网络建模网络结构
- ③ 端到端训练

Unsupervised Deep Embedding for Clustering Analysis (DEC)

DEC³是最具代表性的深度聚类算法之一。它将样本表征的学习和聚类分配的学习统一到了一个框架中，共同学习。



³Unsupervised Deep Embedding for Clustering Analysis(ICML 2016)

DEC——基本步骤

- ① 初始化网络参数 θ
- ② 输入样本 x , 提取其特征 $z = f_{\theta}(x)$
- ③ 根据特征 z 和聚类中心 μ 计算样本的软分配 q
- ④ 根据软分配 q 构造目标分配 p
- ⑤ 根据软分配和目标分配计算 loss, 回传更新网络参数和聚类中心
- ⑥ 重复步骤 2 ~ 5, 直到满足退出条件



DEC——目标函数

根据特征 z 和聚类中心 μ 计算样本的软分配 q :

$$q_{ij} = \frac{(1 + \|z_i - \mu_j\|^2 / \alpha)^{-\frac{\alpha+1}{2}}}{\sum_{j'} (1 + \|z_i - \mu_{j'}\|^2 / \alpha)^{-\frac{\alpha+1}{2}}}$$

其中 α 为 Student's t 分布的自由度, DEC 中取 $\alpha = 1$ 。

DEC 假设特征与聚类中心之间的距离 $\|z - \mu\|^2$ 满足 Student's t 分布, 即把 $\|z - \mu\|^2$ 当做 Student's t 分布中的 x 。

而前面的系数 $\frac{\Gamma(\frac{\alpha+1}{2})}{\sqrt{\alpha}\pi\Gamma(\frac{\alpha}{2})}$ 是一个关于 α 的值, 可以在分子和分母中上下约掉。

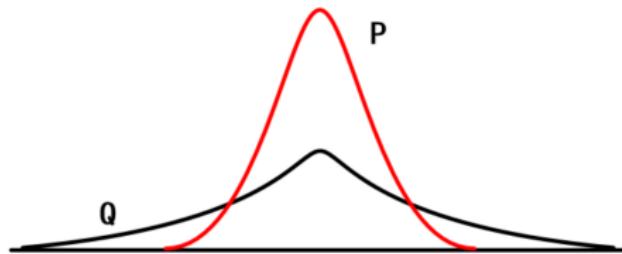


DEC——目标函数

- 根据软分配 q 构造目标分配 p :

$$p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_{j'} q_{ij'}^2 / f_{j'}}$$

其中 $f_j = \sum_i q_{ij}$ 是软聚类频率，有利于类别平衡，避免平凡解。



而 q^2 的作用是使软分配 q 中概率大的在目标分配 p 越大，小的越小。



DEC——目标函数

Loss 函数：

$$L = KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

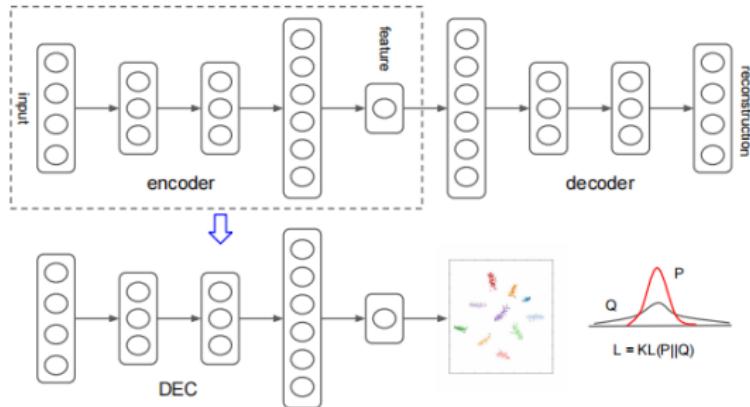
KL 散度又称相对熵，可以用来衡量两个分布之间的差异，两者差异越小，KL 散度越小。当两分布一致时，其 KL 散度为 0。

$$\begin{aligned} D_{KL}(p\|q) &= H(p, q) - H(p) \\ &= - \sum_x p(x) \log q(x) - \sum_x -p(x) \log p(x) \\ &= - \sum_x p(x) \log \frac{q(x)}{p(x)} \end{aligned}$$

选用 KL 散度作为 Loss 函数是希望软分配 q 尽可能向目标分配 p 靠拢。



DEC 总结



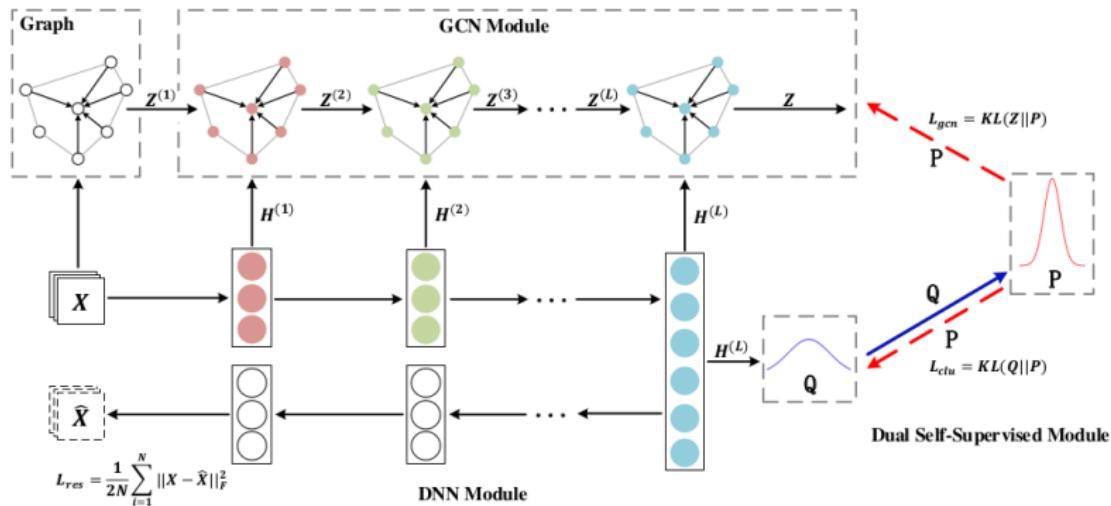
DEC 的优势

- ① 将深度表征学习引入聚类中
- ② 构造了自训练目标函数，引导聚类学习
- ③ 框架通用，可用于不同的数据类型

Structural Deep Clustering Network(SDCN)

研究动机

SDCN 将 DEC 的思路进一步扩展到图数据上，将离散数据的深度聚类思想应用到图数据的社区发现。



研究动机

可用于社区发现的信息包含：

- ① 节点的属性
- ② 网络拓扑结构
- ③ 融合节点属性和网络拓扑结构

节点的属性：自编码器结构

$$\mathcal{L}_{res} = \frac{1}{2N} \sum_{i=1}^N \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 = \frac{1}{2N} \|\mathbf{X} - \hat{\mathbf{X}}\|_F^2$$



网络拓扑结构编码与融合：

多层 GCN 模块：

$$\mathbf{Z}^{(\ell)} = \phi \left(\widetilde{\mathbf{D}}^{-\frac{1}{2}} \widetilde{\mathbf{A}} \widetilde{\mathbf{D}}^{-\frac{1}{2}} \widetilde{\mathbf{Z}}^{(\ell-1)} \mathbf{W}^{(\ell-1)} \right)$$

第 ℓ 层卷积层的输出 $\mathbf{Z}^{(\ell)}$ 与第 ℓ 层编码器的输出 $\mathbf{H}^{(\ell)}$ 线性融合：

$$\widetilde{\mathbf{Z}}^{(\ell)} = (1 - \epsilon) \mathbf{Z}^{(\ell)} + \epsilon \mathbf{H}^{(\ell)}$$

最后一层卷积层的输出作为 Softmax 多分类层：

$$Z = \text{softmax} \left(\widetilde{\mathbf{D}}^{-\frac{1}{2}} \widetilde{\mathbf{A}} \widetilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{Z}^{(L)} \mathbf{W}^{(L)} \right)$$



节点转化为特征空间中的向量后，社区发现转化为了节点表征空间中的聚类问题：

$$q_{ij} = \frac{\left(1 + \|\mathbf{h}_i - \boldsymbol{\mu}_j\|^2 / v\right)^{-\frac{v+1}{2}}}{\sum_{j'} \left(1 + \|\mathbf{h}_i - \boldsymbol{\mu}_{j'}\|^2 / v\right)^{-\frac{v+1}{2}}}$$

$$p_{ij} = \frac{q_{ij}^2 / f_j}{\sum_{j'} q_{ij'}^2 / f_{j'}}$$

$$f_j = \sum_i q_{ij}$$

SDCN-损失函数

- ① 第一部分：解码器的重构特征与原始特征的误差。

$$\mathcal{L}_{res} = \frac{1}{2N} \sum_{i=1}^N \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 = \frac{1}{2N} \|\mathbf{X} - \hat{\mathbf{X}}\|_F^2$$

- ② 第二部分：目标分布 P 通过最小化 Q 和 P 分布之间的 KL 散度损失。

$$\mathcal{L}_{clu} = KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

- ③ 第三部分：GCN 模块也提供一个聚类分布 Z ，因此可以使用分布 P 对分布 Z 进行监督。

$$\mathcal{L}_{gcn} = KL(P\|Z) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{z_{ij}}$$



SDCN-算法流程描述

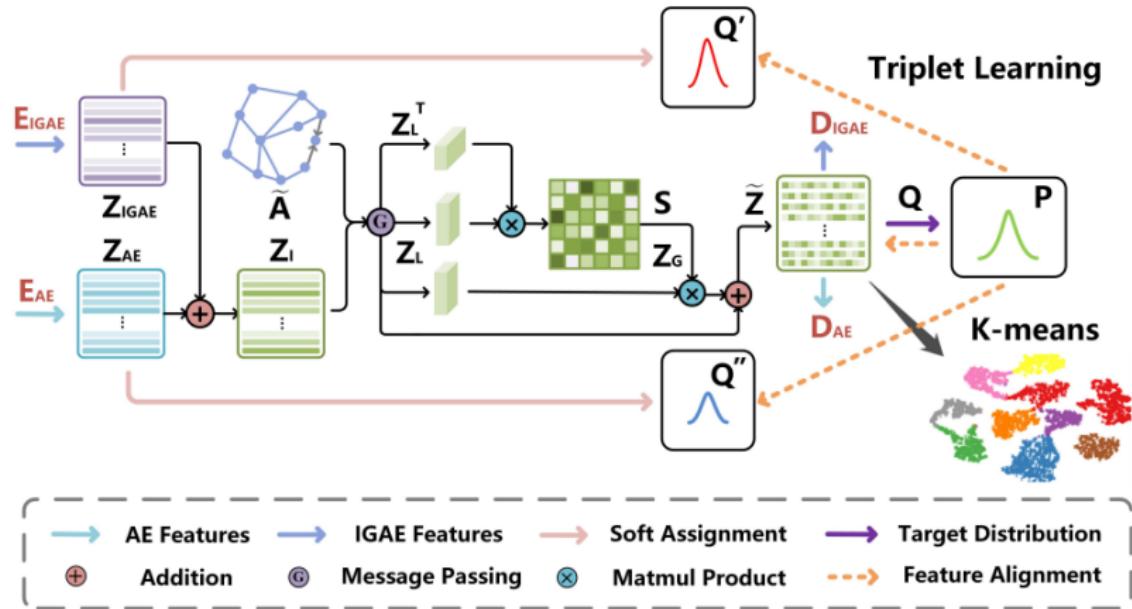
- ① 初始化编码器和解码器的参数，并对其进行预训练。
- ② 使用已训练好的编码器生成表征 \mathbf{H}^L ，在表征上进行 K-means 聚类获得聚类中心 \mathbf{u} 。
- ③ 随机初始化 GCN 模块的参数。
- ④ 使用编码器生成表征 $\mathbf{H}^{(1)}, \mathbf{H}^{(2)}, \dots, \mathbf{H}^{(L)}$
- ⑤ 使用传递算子把第 ℓ 层编码器生成的表征 \mathbf{H}^ℓ 与第 ℓ 层卷积层生成的表征融合并作为第 $(\ell + 1)$ 层卷积层的输入。并在 GCN 模块上进行 L 次迭代生成最后的表征 \mathbf{Z} 。
- ⑥ 将 $\mathbf{H}^{(L)}$ 作为解码器的输入来获取重构表征 $\widehat{\mathbf{X}}$ 。
- ⑦ 计算损失。
- ⑧ 反向传播优化参数。
- ⑨ 重复步骤 4-8。



SDCN-实验结果

Dataset	Metric	K-means	AE	DEC	IDEC	GAE	VGAE	DAEGC	SDCN _Q	SDCN
USPS	ACC	66.82±0.04	71.04±0.03	73.31±0.17	76.22±0.12*	63.10±0.33	56.19±0.72	73.55±0.40	77.09±0.21	78.08±0.19
	NMI	62.63±0.05	67.53±0.03	70.58±0.25	75.56±0.06*	60.69±0.58	51.08±0.37	71.12±0.24	77.71±0.21	79.51±0.27
	ARI	54.55±0.06	58.83±0.05	63.70±0.27	67.86±0.12*	50.30±0.55	40.96±0.59	63.33±0.34	70.18±0.22	71.84±0.24
	F1	64.78±0.03	69.74±0.03	71.82±0.21	74.63±0.10*	61.84±0.43	53.63±1.05	72.45±0.49	75.88±0.17	76.98±0.18
HHAR	ACC	59.98±0.02	68.69±0.31	69.39±0.25	71.05±0.36	62.33±1.01	71.30±0.36	76.51±2.19*	83.46±0.23	84.26±0.17
	NMI	58.86±0.01	71.42±0.97	72.91±0.39	74.19±0.39*	55.06±1.39	62.95±0.36	69.10±2.28	78.82±0.28	79.90±0.09
	ARI	46.09±0.02	60.36±0.88	61.25±0.51	62.83±0.45*	42.63±1.63	51.47±0.73	60.38±2.15	71.75±0.23	72.84±0.09
	F1	58.33±0.03	66.36±0.34	67.29±0.29	68.63±0.33	62.64±0.97	71.55±0.29	76.89±2.18*	81.45±0.14	82.58±0.08
Reuters	ACC	54.04±0.01	74.90±0.21	73.58±0.13	75.43±0.14*	54.40±0.27	60.85±0.23	65.50±0.13	79.30±0.11	77.15±0.21
	NMI	41.54±0.51	49.69±0.29	47.50±0.34	50.28±0.17*	25.92±0.41	25.51±0.22	30.55±0.29	56.89±0.27	50.82±0.21
	ARI	27.95±0.38	49.55±0.37	48.44±0.14	51.26±0.21*	19.61±0.22	26.18±0.36	31.12±0.18	59.58±0.32	55.36±0.37
	F1	41.28±2.43	60.96±0.22	64.25±0.22*	63.21±0.12	43.53±0.42	57.14±0.17	61.82±0.13	66.15±0.15	65.48±0.08
ACM	ACC	67.31±0.71	81.83±0.08	84.33±0.76	85.12±0.52	84.52±1.44	84.13±0.22	86.94±2.83*	86.95±0.08	90.45±0.18
	NMI	32.44±0.46	49.30±0.16	54.54±1.51	56.61±1.16*	55.38±1.92	53.20±0.52	56.18±4.15	58.90±0.17	68.31±0.25
	ARI	30.60±0.69	54.64±0.16	60.64±1.87	62.16±1.50*	59.46±3.10	57.72±0.67	59.35±3.89	65.25±0.19	73.91±0.40
	F1	67.57±0.74	82.01±0.08	84.51±0.74	85.11±0.48	84.65±1.33	84.17±0.23	87.07±2.79*	86.84±0.09	90.42±0.19
DBLP	ACC	38.65±0.65	51.43±0.35	58.16±0.56	60.31±0.62	61.21±1.22	58.59±0.06	62.05±0.48*	65.74±1.34	68.05±1.81
	NMI	11.45±0.38	25.40±0.16	29.51±0.28	31.17±0.50	30.80±0.91	26.92±0.06	32.49±0.45*	35.11±1.05	39.50±1.34
	ARI	6.97±0.39	12.21±0.43	23.92±0.39	25.37±0.60*	22.02±1.40	17.92±0.07	21.03±0.52	34.00±1.76	39.15±2.01
	F1	31.92±0.27	52.53±0.36	59.38±0.51	61.33±0.56	61.41±2.23	58.69±0.07	61.75±0.67*	65.78±1.22	67.71±1.51
Citeseer	ACC	39.32±3.17	57.08±0.13	55.89±0.20	60.49±1.42	61.35±0.80	60.97±0.36	64.54±1.39*	61.67±1.05	65.96±0.31
	NMI	16.94±3.22	27.64±0.08	28.34±0.30	27.17±2.40	34.63±0.65	32.69±0.27	36.41±0.86*	34.39±1.22	38.71±0.32
	ARI	13.43±3.02	29.31±0.14	28.12±0.36	25.70±2.65	33.55±1.18	33.13±0.53	37.78±1.24*	35.50±1.49	40.17±0.43
	F1	36.08±3.53	53.80±0.11	52.62±0.17	61.62±1.39	57.36±0.82	57.70±0.49	62.20±1.32*	57.82±0.98	63.62±0.24

Deep Fusion Clustering Network(DFCN)



DFCN 框架⁴

⁴Deep Fusion Clustering Network(AAAI 2021)

IGAE

针对属性自编码器难以捕获高阶信息以及 GCN 容易过平滑等问题，将两种信息融合后希望能同时重建邻接矩阵和属性矩阵以提升表征的表达能力。

GCN 表征学习：

$$\mathbf{Z}^{(l)} = \sigma \left(\tilde{\mathbf{A}} \mathbf{Z}^{(l-1)} \mathbf{W}^{(l)} \right)$$

重构邻接矩阵：

$$\hat{\mathbf{A}} = \sigma \left(\mathbf{Z} \mathbf{Z}^{\top} \right)$$

$$L_a = \frac{1}{2N} \| \tilde{\mathbf{A}} - \hat{\mathbf{A}} \|_F^2$$



DFCN——IGAE 损失函数

重构属性权重矩阵：

$$\widehat{\mathbf{Z}}^{(h)} = \sigma \left(\widetilde{\mathbf{A}} \widehat{\mathbf{Z}}^{(h-1)} \widehat{\mathbf{W}}^{(h)} \right)$$

$$L_w = \frac{1}{2N} \|\widetilde{\mathbf{A}}\mathbf{X} - \widehat{\mathbf{Z}}\|_F^2$$

混合损失函数：

$$L_{IGAE} = L_w + \gamma L_a$$



DFCN——跨模态动态融合机制

为了增强节点属性和网络结构之间的融合，DFCN 提出了包含更多层融合的跨模态动态融合机制：

- ① 属性和网络结构的表征融合：

$$\mathbf{Z}_I = \alpha \mathbf{Z}_{AE} + (1 - \alpha) \mathbf{Z}_{IGAE}$$

- ② 局部结构融合：

$$\mathbf{Z}_L = \tilde{\mathbf{A}} \mathbf{Z}_I$$

- ③ 全局结构融合：

$$\mathbf{S}_{ij} = \frac{e^{(\mathbf{z}_L \mathbf{z}_L^T)_{ij}}}{\sum_{k=1}^N e^{(\mathbf{z}_L \mathbf{z}_L^T)_{ik}}}$$

$$\mathbf{Z}_G = \mathbf{S} \mathbf{Z}_L$$

- ④ 局部全局整合

$$\tilde{\mathbf{Z}} = \beta \mathbf{Z}_G + \mathbf{Z}_L$$



DFCN——三重自训练学习策略

基于自训练的聚类优化：

$$q_{ij} = \frac{\left(1 + \|\tilde{z}_i - u_j\|^2 / v\right)^{-\frac{v+1}{2}}}{\sum_{j'} \left(1 + \|\tilde{z}_i - u_{j'}\|^2 / v\right)^{-\frac{v+1}{2}}}$$

$$p_{ij} = \frac{q_{ij}^2 / \sum_i q_{ij}}{\sum_{j'} \left(q_{ij'}^2 / \sum_i q_{ij'}\right)}$$

基于 IGAE, AE, SAIF 的三重自训练策略：

$$L_{KL} = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{(q_{ij} + q'_{ij} + q''_{ij}) / 3}$$



DFCN——主要算法流程

- ① 预训练 AE,IGAE,SAIF 初始化网络参数，得到 Z_{AE}, Z_{IGAE} 和共识表征 \tilde{Z}
- ② 使用 K-means 算法对共识表征进行聚类，初始化聚类中心
- ③ 根据跨模态动态融合机制更新节点表征 $\tilde{Z}, Z_{AE}, Z_{IGAE}$
- ④ 根据节点表征 $\tilde{Z}, Z_{AE}, Z_{IGAE}$ 和聚类中心 μ 计算样本的软分配 Q, Q', Q''
- ⑤ 根据软分配 q 构造目标分配 p
- ⑥ 根据软分配和目标分配计算三重聚类损失
- ⑦ 计算联合损失函数，回传更新网络参数和聚类中心
- ⑧ 重复步骤 3 ~ 7，直到满足退出条件



DFCN——实验结果

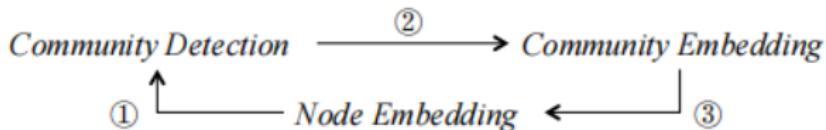
DFCN 实验结果

Data	Metric	K-means	AE	DEC	IDEC	GAE	VGAE	ARGA	DAEGC	SDCN _Q	SDCN	DFCN
USPS	ACC	66.8±0.0	71.0±0.0	73.3±0.2	76.2±0.1	63.1±0.3	56.2±0.7	66.8±0.7	73.6±0.4	77.1±0.2	78.1±0.2	79.5±0.2
	NMI	62.6±0.0	67.5±0.0	70.6±0.3	75.6±0.1	60.7±0.6	51.1±0.4	61.6±0.3	71.1±0.2	77.7±0.2	79.5±0.3	82.8±0.3
	ARI	54.6±0.0	58.8±0.1	63.7±0.3	67.9±0.1	50.3±0.6	41.0±0.6	51.1±0.6	63.3±0.3	70.2±0.2	71.8±0.2	75.3±0.2
	F1	64.8±0.0	69.7±0.0	71.8±0.2	74.6±0.1	61.8±0.4	53.6±1.1	66.1±1.2	72.5±0.5	75.9±0.2	77.0±0.2	78.3±0.2
HIHAR	ACC	60.0±0.0	68.7±0.3	69.4±0.3	71.1±0.4	62.3±1.0	71.3±0.4	63.3±0.8	76.5±2.2	83.5±0.2	84.3±0.2	87.1±0.1
	NMI	58.9±0.0	71.4±1.0	72.9±0.4	74.2±0.4	55.1±1.4	63.0±0.4	57.1±1.4	69.1±2.3	78.8±0.3	79.9±0.1	82.2±0.1
	ARI	46.1±0.0	60.4±0.9	61.3±0.5	62.8±0.5	42.6±1.6	51.5±0.7	44.7±1.0	60.4±2.2	71.8±0.2	72.8±0.1	76.4±0.1
	F1	58.3±0.0	66.4±0.3	67.3±0.3	68.6±0.3	62.6±1.0	71.6±0.3	61.1±0.9	76.9±2.2	81.5±0.1	82.6±0.1	87.3±0.1
REUT	ACC	54.0±0.0	74.9±0.2	73.6±0.1	75.4±0.1	54.4±0.3	60.9±0.2	56.2±0.2	65.6±0.1	79.3±0.1	77.2±0.2	77.7±0.2
	NMI	41.5±0.5	49.7±0.3	47.5±0.3	50.3±0.2	25.9±0.4	25.5±0.2	28.7±0.3	30.6±0.3	56.9±0.3	50.8±0.2	59.9±0.4
	ARI	28.0±0.4	49.6±0.4	48.4±0.1	51.3±0.2	19.6±0.2	26.2±0.4	24.5±0.4	31.1±0.2	59.6±0.3	55.4±0.4	59.8±0.4
	F1	41.3±2.4	61.0±0.2	64.3±0.2	63.2±0.1	43.5±0.4	57.1±0.2	51.1±0.2	61.8±0.1	66.2±0.2	65.5±0.1	69.6±0.1
ACM	ACC	67.3±0.7	81.8±0.1	84.3±0.8	85.1±0.5	84.5±1.4	84.1±0.2	86.1±1.2	86.9±2.8	87.0±0.1	90.5±0.2	90.9±0.2
	NMI	32.4±0.5	49.3±0.2	54.5±1.5	56.6±1.2	55.4±1.9	53.2±0.5	55.7±1.4	56.2±4.2	58.9±0.2	68.3±0.3	69.4±0.4
	ARI	30.6±0.7	54.6±0.2	60.6±1.9	62.2±1.5	59.5±3.1	57.7±0.7	62.9±2.1	59.4±3.9	65.3±0.2	73.9±0.4	74.9±0.4
	F1	67.6±0.7	82.0±0.1	84.5±0.7	85.1±0.5	84.7±1.3	84.2±0.2	86.1±1.2	87.1±2.8	86.8±0.1	90.4±0.2	90.8±0.2
DBLP	ACC	38.7±0.7	51.4±0.4	58.2±0.6	60.3±0.6	61.2±1.2	58.6±0.1	61.6±1.0	62.1±0.5	65.7±1.3	68.1±1.8	76.0±0.8
	NMI	11.5±0.4	25.4±0.2	29.5±0.3	31.2±0.5	30.8±0.9	26.9±0.1	26.8±1.0	32.5±0.5	35.1±1.1	39.5±1.3	43.7±1.0
	ARI	7.0±0.4	12.2±0.4	23.9±0.4	25.4±0.6	22.0±1.4	17.9±0.1	22.7±0.3	21.0±0.5	34.0±1.8	39.2±2.0	47.0±1.5
	F1	31.9±0.3	52.5±0.4	59.4±0.5	61.3±0.6	61.4±2.2	58.7±0.1	61.8±0.9	61.8±0.7	65.8±1.2	67.7±1.5	75.7±0.8
CITE	ACC	39.3±3.2	57.1±0.1	55.9±0.2	60.5±1.4	61.4±0.8	61.0±0.4	56.9±0.7	64.5±1.4	61.7±1.1	66.0±0.3	69.5±0.2
	NMI	16.9±3.2	27.6±0.1	28.3±0.3	27.2±2.4	34.6±0.7	32.7±0.3	34.5±0.8	36.4±0.9	34.4±1.2	38.7±0.3	43.9±0.2
	ARI	13.4±3.0	29.3±0.1	28.1±0.4	25.7±2.7	33.6±1.2	33.1±0.5	33.4±1.5	37.8±1.2	35.5±1.5	40.2±0.4	45.5±0.3
	F1	36.1±3.5	53.8±0.1	52.6±0.2	61.6±1.4	57.4±0.8	57.7±0.5	54.8±0.8	62.2±1.3	57.8±1.0	63.6±0.2	64.3±0.2

- ① 研究背景
- ② 经典社区发现算法
- ③ 基于自训练的社区发现算法
- ④ 基于迭代优化的社区发现算法
- ⑤ 基于对比学习的社区发现
- ⑥ 社区发现的端到端应用



Learning Community Embedding with Community Detection and Node Embedding on Graphs(ComE)



- ① 社区表征不仅有助于图的可视化等社区级应用，并且有利于社区发现和节点分类。
- ② 节点表征有助于提高社区发现的效果，输出良好的社区来拟合更好的社区表征。
- ③ 社区发现、社区表征和节点表征之间存在一个闭环，可以相互促进。

如何获得节点表征？

基于 DeepWalk 的节点表征学习。

① 一阶临近性

$$O_1 = -\sum_{(v_i, v_j) \in E} \log \sigma(\phi_j^T \phi_i)$$

② 二阶临近性

$$\Delta_{ij} = \log \sigma(\phi_j'^T \phi_i) + \sum_{t=1}^m \mathbb{E}_{v_l \sim P_n(v_l)} [\log \sigma(-\phi_l'^T \phi_i)]$$

$$O_2 = -\alpha \sum_{v_i \in V} \sum_{v_j \in C_i} \Delta_{ij}$$



如何表示一个社区？

将社区表示成一个多维的高斯分布，节点的 embedding 作为该高斯分布的随机变量。第 k 个社区可以表示为：

$$N(\psi_k, \Sigma_k)$$

$\psi_k \in \mathbb{R}^d$ 为社区表征的均值， $\Sigma_k \in \mathbb{R}^{d \times d}$ 为社区表征的协方差矩阵，它们都由上一步社区发现结果中属于社区 k 的节点表征计算得来。

社区表征如何帮助提升节点表征？

相同社区的节点往往具有相似的节点表征向量。

相同社区的成员节点之间可能不属于直接相连，即它们之间的连接有可能是高阶的。

社区表征通过让相同社区的成员节点具有相似的节点表征，拓展了节点表征捕获高阶临近性的能力。



ComE-社区发现和社区嵌入

节点 v_i 的表征 ϕ_i 出现的概率：

$$p(z_i = k) p(v_i | z_i = k; \phi_i, \psi_k, \Sigma_k)$$

通过上式构建最大似然函数：

$$\prod_{i=1}^{|V|} \sum_{k=1}^K p(z_i = k) \cdot p(v_i | z_i = k; \phi_i; \psi_k, \Sigma_k)$$

最大对数似然函数作为优化目标：

$$O_3 = -\frac{\beta}{K} \sum_{i=1}^{|V|} \log \sum_{k=1}^K \pi_{ik} \mathcal{N}(\phi_i | \psi_k, \Sigma_k)$$



ComE-优化过程

首先定义：

$$\gamma_{ik} = \frac{\pi_{ik} \mathcal{N}(\phi_i | \psi_k, \Sigma_k)}{\sum_{k'=1}^K \pi_{ik'} \mathcal{N}(\phi_i | \psi_{k'}, \Sigma_{k'})}$$

$$N_k = \sum_{i=1}^{|V|} \gamma_{ik}$$

接着通过如下三个式子更新参数：

$$\pi_{ik} = \frac{N_k}{|V|}$$

$$\psi_k = \frac{1}{N_k} \sum_{i=1}^{|V|} \gamma_{ik} \phi_i$$

$$\Sigma_k = \frac{1}{N_k} \sum_{i=1}^{|V|} \gamma_{ik} (\phi_i - \psi_k) (\phi_i - \psi_k)^T$$



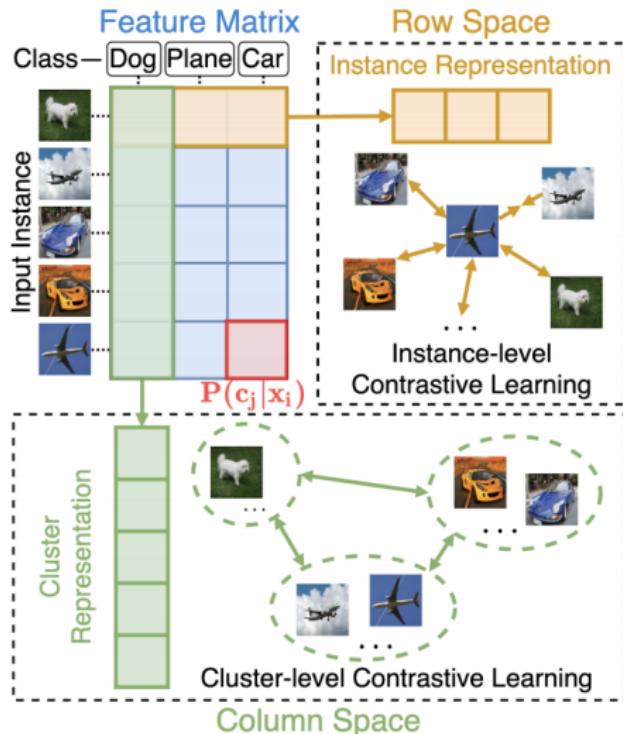
ComE 算法流程概述：

- ① 通过采样获取每个节点的游走路径。
- ② 通过 DeepWalk 算法得到每个节点的嵌入和上下文嵌入。
- ③ 固定 (ϕ, ϕ') ，通过上一页中的三个公式更新参数 (Π, Ψ, Σ) 。
- ④ 固定 (Π, Ψ, Σ) ，通过反向传播优化 (ϕ, ϕ') 。
- ⑤ 循环步骤 3-4。



- ① 研究背景
- ② 经典社区发现算法
- ③ 基于自训练的社区发现算法
- ④ 基于迭代优化的社区发现算法
- ⑤ 基于对比学习的社区发现
- ⑥ 社区发现的端到端应用

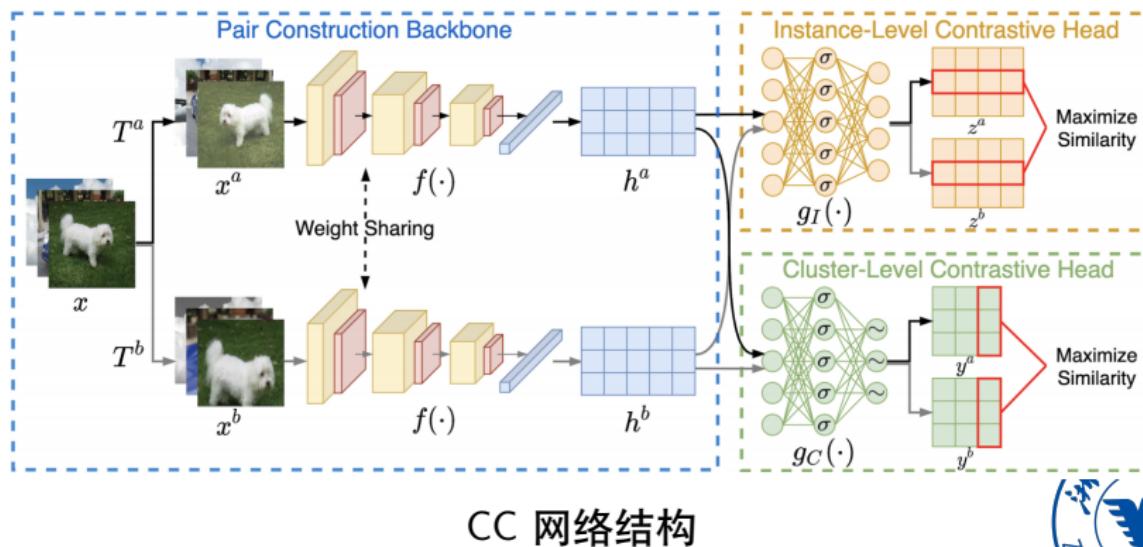




CC 是第一个将对比学习与聚类高度结合的工作。它以单阶段、端到端方式联合学习嵌入表示和集群分配，显式地执行实例级和聚类级的对比学习。因此，它也是一个同步式的深度聚类方法。



CC⁵首先通过数据增强构建数据对，学习相对应的样本特征矩阵和聚类分配矩阵。然后，分别在特征矩阵的行空间和分配矩阵的列空间中进行实例级和聚类级的对比学习。



⁵Contrastive clustering.(AAAI 2021)



CC——实例级对比学习

在实例级对比学习中，CC 对特征矩阵的行向量 z 进行对比学习。

首先定义相似度为 cosine 距离：

$$s(z_i^{k_1}, z_j^{k_2}) = \frac{(z_i^{k_1})(z_j^{k_2})^T}{\|z_i^{k_1}\| \|z_j^{k_2}\|}$$

其中 $k_1, k_2 \in \{a, b\}$ 表示 a 分支和 b 分支， $i, j \in [1, N]$ 。

定义同一样本 x_i 的不同视图 $\{z_i^a, z_i^b\}$ 为正样本对，其余的均为负样本对。



CC——实例级对比学习

对于一个 a 分支的样本 x_i^a , 它的特征向量为 z_i^a , 对应的实例级对比 Loss 为:

$$\ell_i^a = -\log \frac{\exp(s(z_i^a, z_i^b)/\tau_I)}{\sum_{j=1}^N [\exp(s(z_i^a, z_j^a)/\tau_I) + \exp(s(z_i^a, z_j^b)/\tau_I)]}$$

其中 τ_I 为实例级的温度参数。

分子部分为正样本对, 而分母部分则分别是 a 分支中的负样本和 b 分支中的负样本。总的实例级对比学习 Loss 需要考虑两个分支的所有样本, 即:

$$\mathcal{L}_{ins} = \frac{1}{2N} \sum_{i=1}^N (\ell_i^a + \ell_i^b)$$



CC——聚类级对比学习

聚类级的对比学习将每个类看作一个独立的实例，再进行实例级对比学习。

对于一个 batch 的样本，可以得到 a 分支下的分配矩阵 $Y^a \in \mathcal{R}^{N \times M}$ ，其中 N 为 batch 的大小， M 为簇的数量。

从聚类的角度来看这个分配矩阵：共有 M 列，每一列 $y_i^a (i = 1, 2, \dots, M)$ 可以看做为一个类的特征。

由此，相应的正样本对即为 $\{y_i^a, y_j^b\}$ ，其余都为负样本对。



CC——聚类级对比学习

同样的，CC 用 cosine 距离定义类对的相似度：

$$s(y_i^{k_1}, y_j^{k_2}) = \frac{(y_i^{k_1})^T (y_j^{k_2})}{\|y_i^{k_1}\| \|y_j^{k_2}\|}$$

其中 $k_1, k_2 \in \{a, b\}$, $i, j \in [1, M]$ 。

对于 a 分支下的类 i , 它的特征为 y_i^a , 对应的聚类级对比 Loss 为：

$$\hat{\ell}_i^a = -\log \frac{\exp(s(y_i^a, y_i^b)/\tau_C)}{\sum_{j=1}^M [\exp(s(y_i^a, y_j^a)/\tau_C) + \exp(s(y_i^a, y_j^b)/\tau_C)]}$$

其中 τ_C 为聚类级温度参数。

CC——目标函数

同样，总的聚类级对比 Loss 需要考虑两个分支内所有的类：

$$\mathcal{L}_{clu} = \frac{1}{2M} \sum_{i=1}^M (\hat{\ell}_i^a + \hat{\ell}_i^b) - H(Y)$$

其中， $H(Y)$ 为 Y 矩阵的负熵，这一部分能够帮助避免平凡解。

$$H(Y) = \sum_{i=1}^M [P(y_i^a) \log(P(y_i^a)) + P(y_i^b) \log(P(y_i^b))]$$

$$P(y_i^k) = \sum_{j=1}^N \frac{Y_{ji}^k}{\|Y\|_1}, \quad k \in \{a, b\}$$



Efficient Graph Convolution for Joint Node Representation Learning and Clustering(GCC)

研究动机

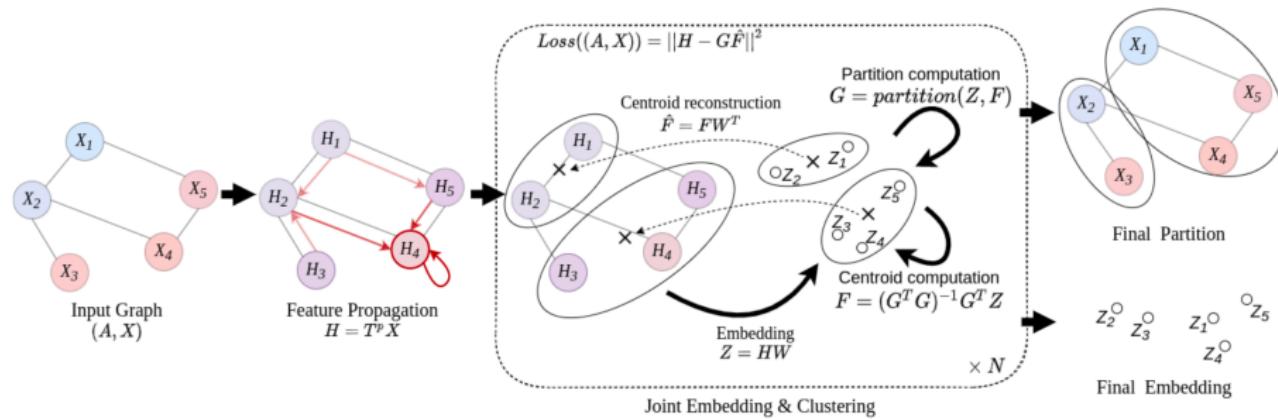
现有方法多是通过学习很好的节点表征，并将聚类作为下游任务处理，而通过将聚类目标纳入表征学习过程可以获得更好的性能。

GCC⁶解决方案：

- 联合表征学习和聚类能过取得更好的聚类性能。
- 最小化节点属性重构损失，并通过节点表征与聚类中心的距离来产生聚类友好的表征。

⁶Efficient Graph Convolution for Joint Node Representation Learning and Clustering(WSDM 2022)

GCC: framework



GCC: Joint Graph Representation Learning and Clustering

联合表征学习和聚类

最小化节点属性重构损失，同时最小化节点表征与聚类中心的距离。

$$\begin{aligned} & \min_{\theta_1, \theta_2, \mathbf{G}, \mathbf{F}} \underbrace{\| \text{dec}_{\theta_2} (\text{enc}_{\theta_1}(\text{agg}(\mathbf{A}, \mathbf{X}))) - \text{agg}(\mathbf{A}, \mathbf{X}) \|^2}_{\text{reconstruction term}} \\ & + \alpha \underbrace{\| \text{enc}_{\theta_1}(\text{agg}(\mathbf{A}, \mathbf{X})) - \mathbf{GF} \|^2}_{\text{clustering regularization term}} \\ & \text{s.t. } \mathbf{G} \in \{0, 1\}^{n \times k}, \mathbf{G}\mathbf{1}_k = \mathbf{1}_n \end{aligned}$$



GCC: Normalized Simple Graph Convolution

定义归一化简单图卷积

$$\text{agg}(\mathbf{A}, \mathbf{X}) = \mathbf{T}^p \mathbf{X}$$

$$\mathbf{T} = \mathbf{D}_{\mathbf{T}}^{-1} \left(\mathbf{I} + \tilde{\mathbf{D}}^{-1/2} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-1/2} \right)$$

联合表征学习和聚类的目标函数进一步定义为

$$\begin{aligned} & \min_{\mathbf{G}, \mathbf{F}, \mathbf{W}_1, \mathbf{W}_2} \| \mathbf{T}^p \mathbf{X} - \mathbf{T}^p \mathbf{X} \mathbf{W}_1 \mathbf{W}_2 \|^2 + \alpha \| \mathbf{T}^p \mathbf{X} \mathbf{W}_1 - \mathbf{G} \mathbf{F} \|^2 \\ & \text{s.t. } \mathbf{G} \in \{0, 1\}^{n \times k}, \mathbf{G} \mathbf{1}_k = \mathbf{1}_n \end{aligned}$$



GCC: 实验结果

Method	Input	Citeseer			Cora			Pubmed			Wiki		
		Acc	F1	NMI									
Sph. k-means	X	42.64	40.16	19.91	33.97	30.93	15.33	59.51	58.16	31.26	33.65	23.30	29.90
DCN	X	19.16	11.44	2.91	20.01	11.81	2.32	15.87	7.06	4.07	44.28	17.14	12.45
Spectral	A	21.60	9.46	1.54	30.00	8.78	2.36	58.96	43.53	18.30	23.20	13.74	18.05
LAE (2020)	(A, X)	43.49	41.33	22.66	65.43	66.21	48.89	OOM			45.26	40.90	45.99
LVAE (2020)	(A, X)	39.46	38.26	20.53	64.11	65.31	48.47	OOM			47.38	42.92	47.79
AGE (2020)	(A, X)	57.85	55.01	35.74	69.17	67.30	56.91	OOM			53.79	41.39	52.63
GIC (2021)	(A, X)	68.78	64.02	43.82	70.45	68.95	52.55	64.30	64.86	26.02	46.46	40.29	48.24
S ² GC (2021)	(A, X)	68.13	63.79	42.26	69.68	66.41	54.83	70.81	69.96	32.32	52.71	44.40	48.96
GCC (ours)	(A, X)	69.45	64.54	45.13	74.29	70.35	59.17	70.82	69.89	32.30	54.56	46.10	54.61



Deep Graph Clustering via Dual Correlation Reduction(DCRN)

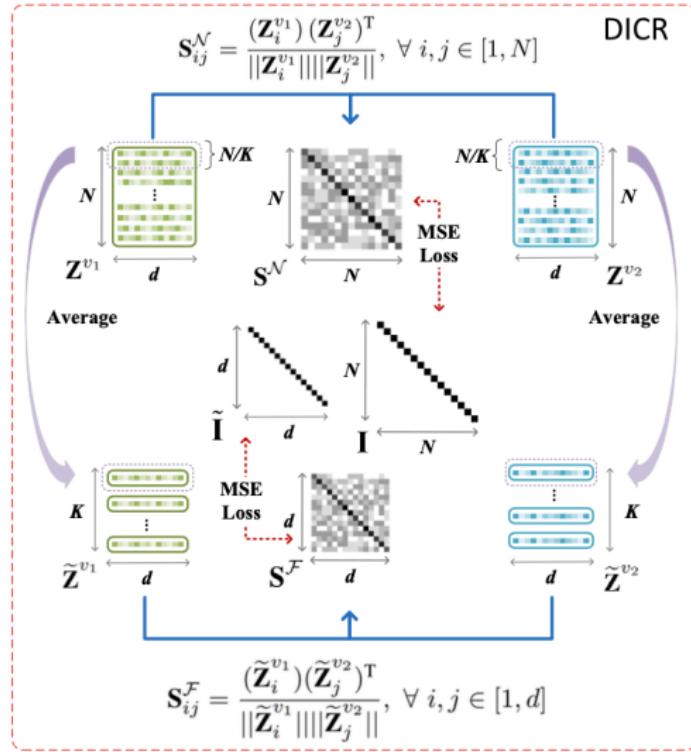
研究动机

考虑到 Representation Collapse 使得表征判别力下降，从而聚类性能不佳，文章提出利用 dual information correlation reduction (DICR) 来提升表征判别力。

Representation Collapse

模型对于所有节点输出非常相似的 representation，没有很好捕获图上类内相似、类间不同的特性。比如，在 contrastive learning 中，仅有 alignment 约束而没有表征空间 uniformity 约束会导致 representation collapse。

DCRN: DICR 原理



Dual Information Correlation Reduction



DCRN: DICR 原理

(1) 构造 sample-level correlation reduction (SCR), 计算两个 augmented views 的 embedding 矩阵的相似度, 让同节点趋向 1, 不同节点最小化

$$\mathbf{S}_{ij}^{\mathcal{N}} = \frac{(\mathbf{Z}_i^{v_1}) (\mathbf{Z}_j^{v_2})^T}{\|\mathbf{Z}_i^{v_1}\| \|\mathbf{Z}_j^{v_2}\|}, \forall i, j \in [1, N]$$

$$\begin{aligned}\mathcal{L}_N &= \frac{1}{N^2} \sum (\mathbf{S}^{\mathcal{N}} - \mathbf{I})^2 \\ &= \frac{1}{N} \sum_{i=1}^N (\mathbf{S}_{ii}^{\mathcal{N}} - 1)^2 + \frac{1}{N^2 - N} \sum_{i=1}^N \sum_{j \neq i} (\mathbf{S}_{ij}^{\mathcal{N}})^2\end{aligned}$$



DCRN: DICR 原理

(2) 构造 feature-level correlation reduction (FCR), 将 $\mathbb{R}^{d \times N}$ 的 embedding readout 为 $\mathbb{R}^{d \times K}$, 计算两个 augmented views 的 dimension 相似度矩阵 (类分布相似度), 让同维趋向 1, 不同维最小化

$$\tilde{\mathbf{Z}}^{v_k} = \mathcal{R} \left((\mathbf{Z}^{v_k})^T \right), \text{ where } \mathcal{R}(\cdot) : \mathbb{R}^{d \times N} \rightarrow \mathbb{R}^{d \times K}$$

$$\mathbf{S}_{ij}^{\mathcal{F}} = \frac{\left(\tilde{\mathbf{Z}}_i^{v_1} \right) \left(\tilde{\mathbf{Z}}_j^{v_2} \right)^T}{\left\| \tilde{\mathbf{Z}}_i^{v_1} \right\| \left\| \tilde{\mathbf{Z}}_j^{v_2} \right\|}, \forall i, j \in [1, d],$$

$$\begin{aligned} \mathcal{L}_F &= \frac{1}{d^2} \sum \left(\mathbf{S}^{\mathcal{F}} - \tilde{\mathbf{I}} \right)^2 \\ &= \frac{1}{d^2} \sum_{i=1}^d \left(\mathbf{S}_{ii}^{\mathcal{F}} - 1 \right)^2 + \frac{1}{d^2 - d} \sum_{i=1}^d \sum_{j \neq i} \left(\mathbf{S}_{ij}^{\mathcal{F}} \right)^2 \end{aligned}$$



DCRN: DICR loss

Propagated Regularization:

$$\mathcal{L}_R = JSD(Z, \tilde{A}Z)$$

DICR loss:

$$\mathcal{L}_{DICR} = \mathcal{L}_N + \mathcal{L}_F + \gamma \mathcal{L}_R$$

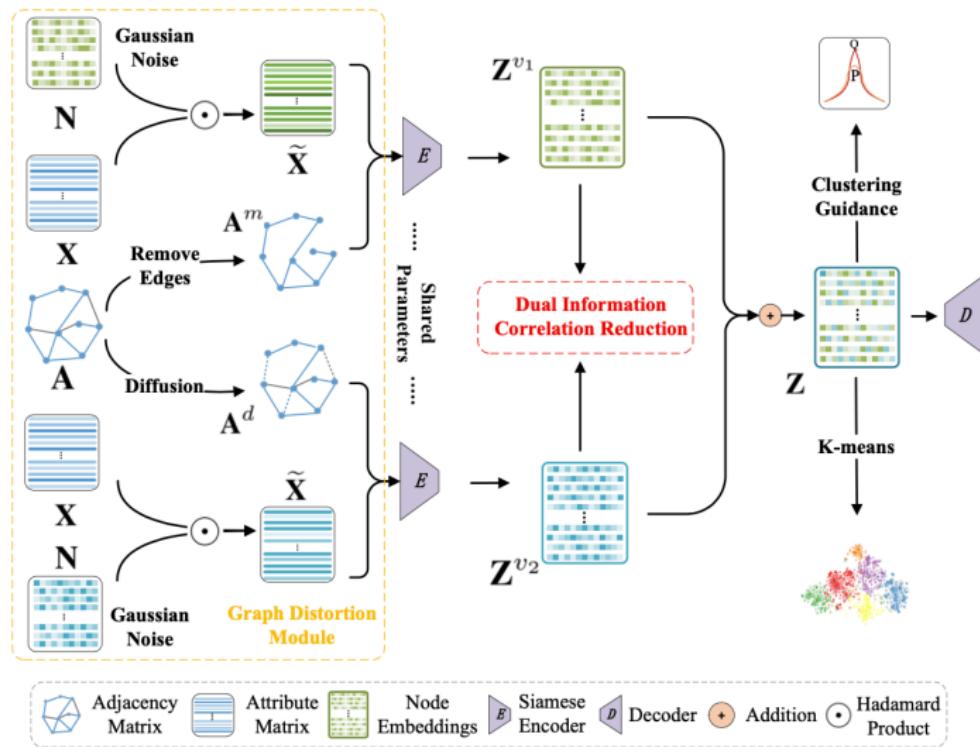
完整训练 loss 如下：

$$\mathcal{L} = \mathcal{L}_{DICR} + \mathcal{L}_{REC} + \lambda \mathcal{L}_{KL}$$

其中 \mathcal{L}_{REC} 表示节点属性和结构的重构损失， \mathcal{L}_{KL} 表示 DGCN 中的自监督 KL 散度损失。



DCRN: 模型图



Dual Correlation Reduction Network (DCRN)



DCRN: 实验结果

Dataset	Metric	K-Means	AE	DEC	IDEC	GAE	DAEGC	ARGA	SDCN	MVGRL	DFCN	Ours
DBLP	ACC	38.65±0.65	51.43±0.35	58.16±0.56	60.31±0.62	61.21±1.22	62.05±0.48	64.83±0.59	68.05±1.81	42.73±1.02	76.00±0.80	79.66±0.25
	NMI	11.45±0.38	25.40±0.16	29.51±0.28	31.17±0.50	30.80±0.91	32.49±0.45	29.42±0.92	39.50±1.34	15.41±0.63	43.70±1.00	48.95±0.44
	ARI	6.97±0.39	12.21±0.43	23.92±0.39	25.37±0.60	22.02±1.40	21.03±0.52	27.99±0.91	39.15±2.01	8.22±0.50	47.00±1.50	53.60±0.46
	F1	31.92±0.27	52.53±0.36	59.38±0.51	61.33±0.56	61.41±2.23	61.75±0.67	64.97±0.66	67.71±1.51	40.52±1.51	75.70±0.80	79.28±0.26
CITE	ACC	39.32±3.17	57.08±0.13	55.89±0.20	60.49±1.42	61.35±0.80	64.54±1.39	61.07±0.49	65.96±0.31	68.66±0.36	69.50±0.20	70.86±0.18
	NMI	16.94±3.22	27.64±0.08	28.34±0.30	27.17±2.40	34.63±0.65	36.41±0.86	34.40±0.71	38.71±0.32	43.66±0.40	43.90±0.20	45.86±0.35
	ARI	13.43±3.02	29.31±0.14	28.12±0.36	25.70±2.65	33.55±1.18	37.78±1.24	34.32±0.70	40.17±0.43	44.27±0.73	45.50±0.30	47.64±0.30
	F1	36.08±3.53	53.80±0.11	52.62±0.17	61.62±1.39	57.36±0.82	62.20±1.32	58.23±0.31	63.62±0.24	63.71±0.39	64.30±0.20	65.83±0.21
ACM	ACC	67.31±0.71	81.83±0.08	84.33±0.76	85.12±0.52	84.52±1.44	86.94±2.83	86.29±0.36	90.45±0.18	86.73±0.76	90.90±0.20	91.93±0.20
	NMI	32.44±0.46	49.30±0.16	54.54±1.51	56.61±1.16	55.38±1.92	56.18±4.15	56.21±0.82	68.31±0.25	60.87±1.40	69.40±0.40	71.56±0.61
	ARI	30.60±0.69	54.64±0.16	60.64±1.87	62.16±1.50	59.46±3.10	59.35±3.89	63.37±0.86	73.91±0.40	65.07±1.76	74.90±0.40	77.56±0.52
	F1	67.57±0.74	82.01±0.08	84.51±0.74	85.11±0.48	84.65±1.33	87.07±2.79	86.31±0.35	90.42±0.19	86.85±0.72	90.80±0.20	91.94±0.20
AMAP	ACC	27.22±0.76	48.25±0.08	47.22±0.08	47.62±0.08	71.57±2.48	76.44±0.01	69.28±2.30	53.44±0.81	45.19±1.79	76.88±0.80	79.94±0.13
	NMI	13.23±1.33	38.76±0.30	37.35±0.05	37.83±0.08	62.13±2.79	65.57±0.03	58.36±2.76	44.85±0.83	36.89±1.31	69.21±1.00	73.70±0.24
	ARI	5.50±0.44	20.80±0.47	18.59±0.04	19.24±0.07	48.82±4.57	59.39±0.02	44.18±4.41	31.21±1.23	18.79±0.47	58.98±0.84	63.69±0.20
	F1	23.96±0.51	47.87±0.20	46.71±0.12	47.20±0.11	68.08±1.76	69.97±0.02	64.30±1.95	50.66±1.49	39.65±2.39	71.58±0.31	73.82±0.12
PUBMED	ACC	59.83±0.01	63.07±0.31	60.14±0.09	60.70±0.34	62.09±0.81	68.73±0.03	65.26±0.12	64.20±1.30	67.01±0.52	68.89±0.07	69.87±0.07
	NMI	31.05±0.02	26.32±0.57	22.44±0.14	23.67±0.29	23.84±3.54	28.26±0.03	24.80±0.17	22.87±2.04	31.59±1.45	31.43±0.13	32.20±0.08
	ARI	28.10±0.01	23.86±0.67	19.55±0.13	20.58±0.39	20.62±1.39	29.84±0.04	24.35±0.17	22.30±2.07	29.42±1.06	30.64±0.11	31.41±0.12
	F1	58.88±0.01	64.01±0.29	61.49±0.10	62.41±0.32	61.37±0.85	68.23±0.02	65.69±0.13	65.01±1.21	67.07±0.36	68.10±0.07	68.94±0.08
CORAFULL	ACC	26.27±1.10	33.12±0.19	31.92±0.45	32.19±0.31	29.60±0.81	34.35±1.00	22.07±0.43	26.67±0.40	31.52±2.95	37.51±0.81	38.80±0.60
	NMI	34.68±0.84	41.53±0.25	41.67±0.24	41.64±0.28	45.82±0.75	49.16±0.73	41.28±0.25	37.38±0.39	48.99±3.95	51.30±0.41	51.91±0.35
	ARI	9.35±0.57	18.13±0.27	16.98±0.29	17.17±0.22	17.84±0.86	22.60±0.47	12.38±0.24	13.63±0.27	19.11±2.63	24.46±0.48	25.25±0.49
	F1	22.57±1.09	28.40±0.30	27.71±0.58	27.72±0.41	25.95±0.75	26.96±1.33	18.85±0.41	22.14±0.43	26.51±2.87	31.22±0.87	31.68±0.76

- ① 研究背景
- ② 经典社区发现算法
- ③ 基于自训练的社区发现算法
- ④ 基于迭代优化的社区发现算法
- ⑤ 基于对比学习的社区发现
- ⑥ 社区发现的端到端应用



社区发现具体可以应用在哪里？

物品推荐

- SoEXBMF 展示了一种基于社区发现的具体应用场景：**物品推荐**
- 任务：为每个用户推荐最可能被他消费的物品

社区发现如何运用于物品推荐？



社区发现与物品推荐的联系

- 用户的消费行为受两方面因素的影响：用户个人的喜好以及对物品的接触（用户是否知道这个物品）
- 许多关于社交推荐的工作都假设有社交关系的用户之间的个人喜好会比较相似
 - 然而，由于社交关系的多样性，这种假设不一定成立
- 比较确定的是，用户对物品的接触会受以下两个因素影响：
 - 社区中其他用户对物品的接触
 - 朋友（有边连接的其他用户）的消费行为



SoEXBMF

- SoEXBMF 主要着眼于社交关系对用户接触物品的影响，不考虑社交关系对个人喜好的影响。
- 在 SoEXBMF 中，物品推荐被分为两个过程：判断用户是否与物品接触，判断用户是否喜欢接触到的物品。
- 社交关系只对用户接触模型产生影响，具体分为：
 - Social knowledge influence：社区中其他用户对物品的接触
 - Social consumption influence：朋友的消费行为

SoEXBMF

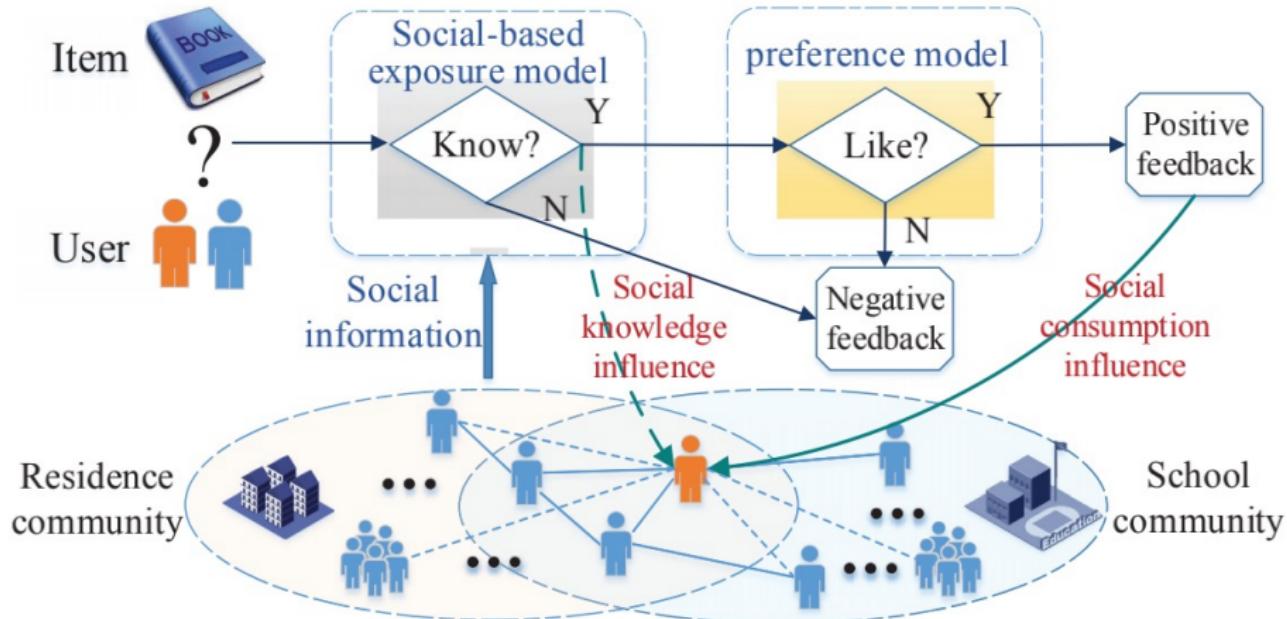
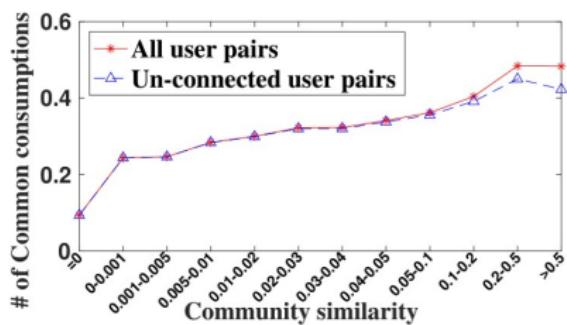


图: SoEXBMF 方法框架

Social knowledge influence

Social knowledge influence

- 物品的信息有可能在社交网络中传播。无论社区中其他用户和我们有没有直接的联系，他们提到的物品都更有可能引起我们的注意。
- 也就是说，**用户对物品的接触会受到其社区中连接和未连接的用户的知识的影响。**



左图中，非直接连接的用户对数量远大于直接连接的用户对。这说明社区内未连接的用户对物品接触也有重要的影响。

公式

形式化地来说，在社交知识的影响下，用户 i 对物品 j 的接触 a_{ij} 服从如下的生成过程：

$$a_{ij} \sim \text{Bernouli}(\sigma(\theta_i^T d_j))$$

其中：

- $\sigma(\cdot)$: logistic 函数，能够将值映射到 $[0, 1]$
- θ_i : 用户 i 的 D 维社区分布。 D 为社区的数量， D 和 θ_i 都由现成的社区发现方法得到
- d_j : 也是一个 D 维向量，表示物品 j 在各个社区中的受欢迎程度

Social consumption influence

Social consumption influence

- 与朋友能接触到的物品相比，**朋友消费的物品显然更容易引起我们的关注。**
 - 当一个用户消费了某个物品，他更有可能向他的朋友分享
 - 电商平台会更倾向于向用户展示他的朋友购买过的物品



公式

在社交消费的影响下，用户 i 对物品 j 的接触 a_{ij} 服从如下的生成过程：

$$a_{ij} \sim \text{Bernouli}\left(\sigma\left(\tau_{i0} + \sum_{k \in \mathcal{T}_i} \tau_{ik} x_{kj}\right)\right)$$

其中：

- \mathcal{T}_i : 用户 i 的朋友集合
- x_{kj} : 用户 k 对物品 j 的消费
- τ_{ik} : 用户 i 受其朋友 k 消费的影响程度



Social-based exposure model

将两种社交影响结合，我们可以得到整个 social-based exposure model 的表达式：

$$a_{ij} \sim \text{Bernouli}\left(\sigma\left(\theta_i^T d_j + \tau_{i0} + \sum_{k \in \mathcal{T}_i} \tau_{ik} x_{kj}\right)\right)$$

preference model

在 preference model 中，模型根据用户是否接触物品 (a) 以及用户的喜好（模型参数）来预测用户是否会消费该物品。

深度社区发现的挑战

- ① 节点标签与社区的关系
- ② 显式的社区约束
- ③ 捕获高阶信息的图神经网络架构
- ④ 类别不均衡问题
- ⑤ 深度社区发现的端到端应用



课程回顾

- ① 研究背景
- ② 经典社区发现算法
- ③ 基于自训练的社区发现算法
- ④ 基于迭代优化的社区发现算法
- ⑤ 基于对比学习的社区发现
- ⑥ 社区发现的端到端应用



References I

-  Chen, J., Feng, Y., Ester, M., Zhou, S., Chen, C., and Wang, C. (2018).

Modeling users' exposure with social knowledge influence and consumption influence for recommendation.

In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 953–962.

