Beijing Forest Studio
北京理工大学信息系统及安全对抗实验中心

GraphGAN

周妍汝　博士研究生

2019年11月03日

内容提要

- 背景简介
- 基本概念
- 算法原理
- 优劣分析
- 应用总结
- 参考文献

背景简介

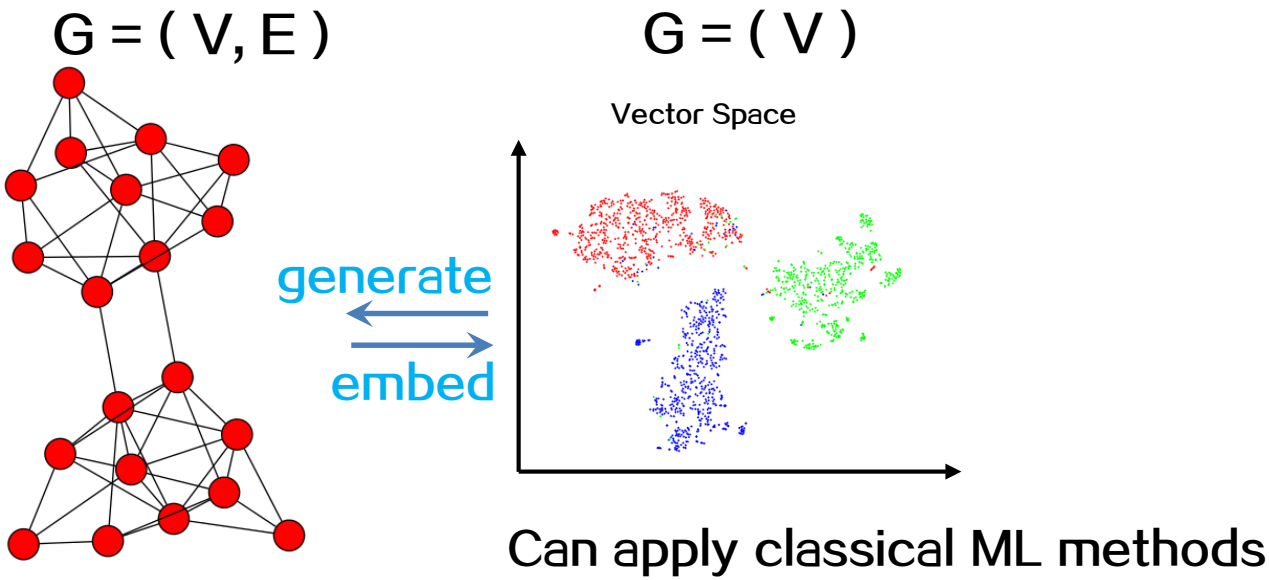
- 预期收获
  - 1. 熟悉图嵌入基本思想
  - 2. 理解GraphGAN的算法原理
  - 3. 了解图嵌入的应用

背景简介

- 图(Graph)是一个非常常用的数据结构：
  - 社交网络/蛋白体结构/交通路网/知识图谱等，甚至规则网格结构数据(如图像，视频等)也是图数据的一种特殊形式，因此图是一个很值得研究的领域。
- Graph研究分类
  - 经典图算法：如生成树算法，最短路算法，复杂一点的二分图匹配，费用流问题等
  - 概率图模型：将条件概率表达为图结构，并进行进一步挖掘，典型的有条件随机场等
  - 图神经网络：研究图结构数据挖掘问题，典型的有Graph Embedding（GE），Graph Convolutional Networks(GCN)等

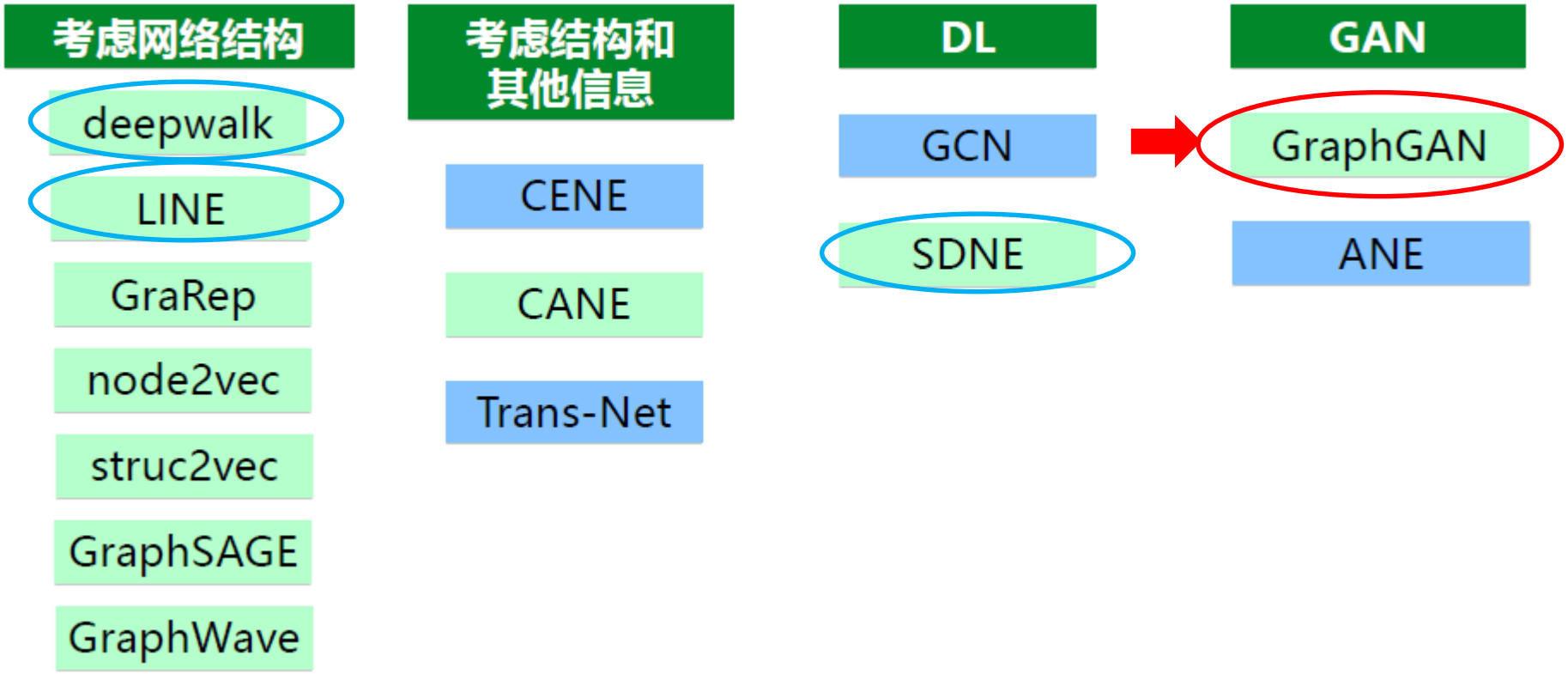
背景简介

- 图嵌入 or 网络表示学习 or 网络嵌入
  - 旨在从网络数据中学习得到网络中每个节点的低维、实值、稠密的向量表示, 之后这些节点表示就可以作为节点的特征应用于后续的网络应用任务中，如节点分类、链接预测、社区发现、可视化任务等。



背景简介

- 有哪些图嵌入方法?



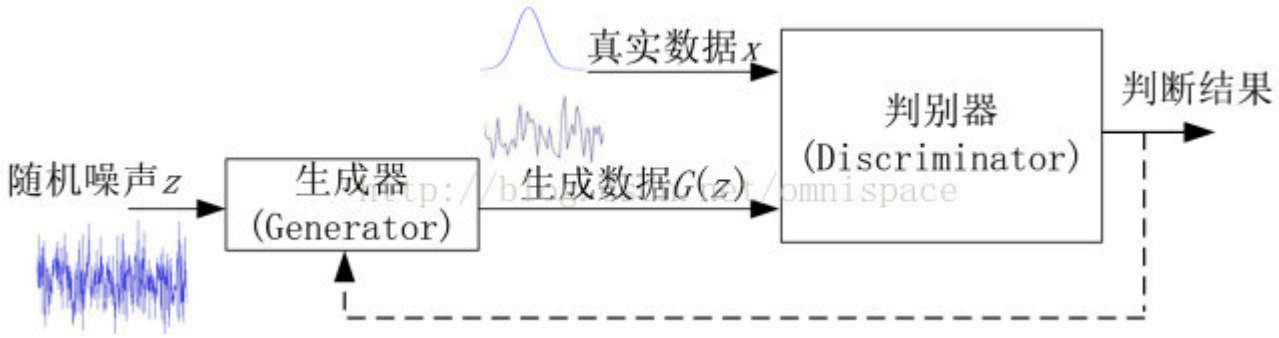
基本概念

基本概念

- Embedding
  - WHAT?
    - Embedding在数学上是一个函数，f : X —> Y，即将一个空间的点映射到另一个空间，通常是从高维抽象的空间映射到低维具象的空间。
    - 一般映射到低维空间的表示具有分布式稠密表示的特性
  - WHY?
    - 抽象的事物应该有一个低维的表示
    - 计算机和神经网络善于处理低维度信息
    - 解决one-hot编码问题
      - one-hot例子：
        - » 祖国特征：["中国"，"美国，"法国"]（这里N=3）：
        - » 中国 => 100；美国 => 010；法国 => 001

基本概念

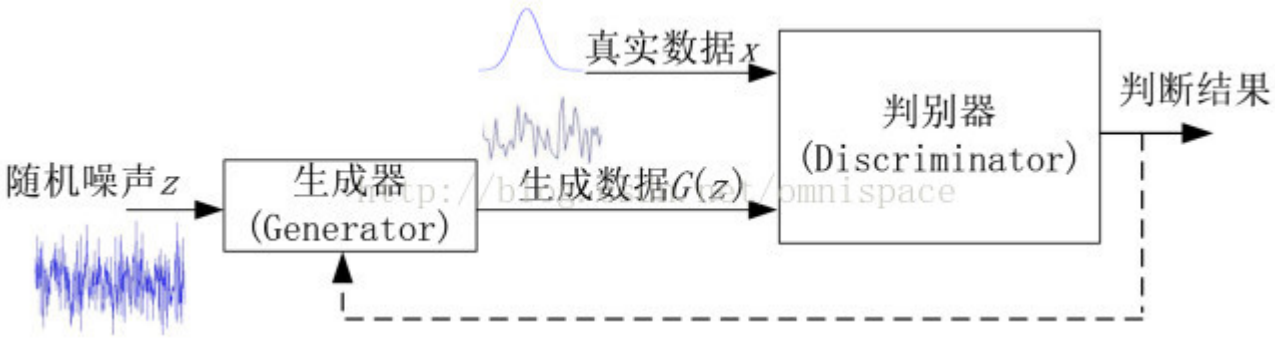
- GAN（Generative Adversarial Network）
  - 生成对抗网络
  - 类比：造假者和警察



详情参考2017年11月份 的学术报告《GAN》——周妍汶

基本概念

- GAN（Generative Adversarial Network）
  - 由两部分主成：生成器G / 判别器D
  - 生成器用一随机噪声向量z尽量去捕捉真实样本数据的分布，判别器是一个二分类器，判别输入是真实数据还是生成的样本。
  - 当对抗过程进行到一定，如果生成器所生成的数据，能够使具有很强分辨能力的判别器仍无法正确判断，生成器实际上已经学到了真实数据的分布。



基本概念

- 优化目标函数

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

  - 其中x表示真实数据，z表示输入生成器G的噪声，G(z)表示G生成的假数据。
  - D(x)表示判别器D判断真实数据x是否真实的概率，$D\big(G(z)\big)$是D判断G生成的假数据的是否真实的概率

基本概念

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

- D的目的:
  - 希望自己的判别能力越强。D的能力越强，D(x)应该越大，$D\big(G(z)\big)$应该越小。这时V(D,G)会变大。因此式子对于D来说是求最大max_D
- G的目的
  - 目标函数的第一项不包含G，是常数，忽略
  - 希望自己生成的数据“越接近真实越好”。G希望$D\big(G(z)\big)$尽可能得大，这时$V(D,G)$会变小。因此式子的最前面的记号是min_G

基本概念

- GAN算法伪代码
  - GAN训练优化的过程实际上是两个模块的相互竞争相互对抗的交替优化过程

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

基本概念

图的符号定义

- $G = (V, E)$表示给定的网络
  - $V = \{v_1, v_2, \cdots, v_V\}$是节点集合，$\mathcal{E} = \{e_{ij}\}_{i,j=1}^{V}$是边的集合，$e_{ij}$表示节点$v_i$和$v_j$之间的边。
- 给定一个节点 v_c
  - $\mathcal{N}(v_c)$表示节点v_c直接相邻的节点（一阶邻居）
  - $p_{true}(v|v_c)$表示的是网络中节点v_c的真实连通性分布条件概率，表示的是节点v_c相对于图中所有其他顶点的连通性偏好（即除了 v_c之外其他节点与v_c之间构成连接边的概率）。
  - 从某个角度上来看，$\mathcal{N}(v_c)$可以看作是基于 $p_{true}(v|v_c)$采样得到的样本集合。

算法原理

算法原理

T	将图中每一个节点都映射到一个低维向量空间
I	输入网络G(V;E)
P	GraphGAN为生成式模型和判别式模型的结合，其包含两个重要部分，即生成器$G(v\|v_c;\theta_G)$和判别器$D(v,v_c;\theta_D)$
O	输出生成器G 和判别器D

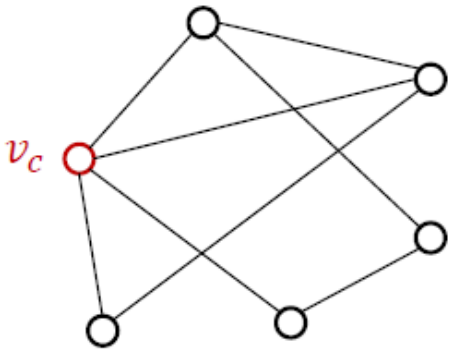
P	将GAN网络的思想应用在图网络特征表达
C	基于GAN模型框架
D	如何保持原有图的结构和距离信息
L	AAAI 2018

算法原理

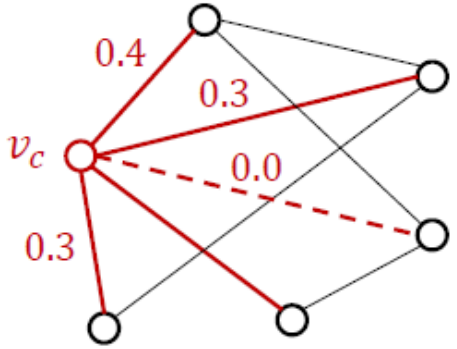
- 将 Graph Embedding 的方法分为两类
  - 生成式方法
  - 判别式方法

算法原理

- 生成式方法
  - 假定对于在图中每个节点v_c存在一个潜在的、真实的连续性分布$p_{true}(v|v_c)$
    - 图中的每条边都可以看作是从$p_{true}(v|v_c)$里采样的一些样本
    - 生成式方法都通过将边的似然概率最大化来学习节点嵌入
    - E.g. DeepWalk（KDD2014）和 Node2vec（KDD2016）



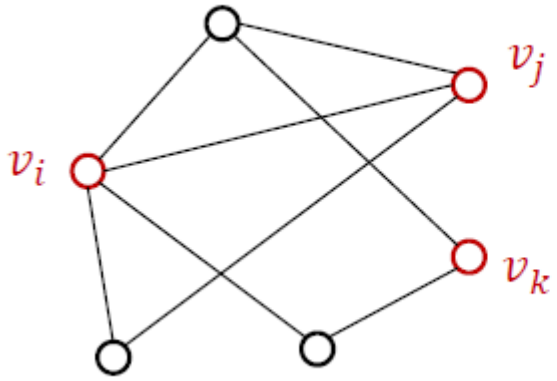
Original graph



$p_{true}(v|v_c)$

算法原理

- 判别式方法
  - 与生成式不同，不将边作为由条件分布$p_{true}(v|v_c)$生成的对象
  - 试图学习一种用于直接预测边缘存在的分类器
    - 将两个节点v_i和v_j共同视为特征，去预测两个节点之间存在边的标量概率$p(edge|v_i, v_j)$
    - E.g. SDNE（KDD 2016）和PPNE（DASFAA 2017）



$$p(edge|v_i, v_j) = 0.8$$
$$p(edge|v_i, v_k) = 0.3$$
……

算法原理

G + D?

- GraphGAN主要有两个模型：
- 生成器$G(v|v_c;\theta_G)$
  - 主要是去尽可能地去拟合或预估出节点v_c真实的连接分布$p_{true}(v|v_c)$，从而在节点集 V 选择出最有可能与 v_c连接的节点
- 判别器$D(v,v_c;\theta_D)$
  - 目的是判断节点v和v_c之间是否有边连接，输出v和v_c存在连接$edge(v,v_c)$的标量概率 $p(edge(v,v_c)|v,v_c)$

算法原理

- 目标函数（two-player minimax game）

$$\min_{\theta_G}\max_{\theta_D} V(G,D)=\sum_{c=1}^{V}(\mathbb{E}_{v\sim p_{true}(\cdot|v_c)}[\log D(v,v_c;\theta_D)]+(\mathbb{E}_{v\sim G(\cdot|v_c;\theta_G)}[\log(1-D(v,v_c;\theta_D))])$$

  - 对图中每个节点的两项期望求和
  - 给定θ_G，优化D：
    - 通过改变 θ_D，判别器D希望自身能够预测准确，即对真实的样本，让$D(v,v_c;\theta_D)_{v\sim p_{true}(\cdot|v_c)}$（第一项）概率值大；对G生成的假样本，让$D(v,v_c;\theta_D)_{v\sim G(v|v_c;\theta_G)}$概率值小，也就是让$\left(1-D(v,v_c;\theta_D)\right)_{v\sim G(v|v_c;\theta_G)}$（第二项）大，因此整体是一个max的目标$max_{\theta_D}V(G,D)$

算法原理

- 目标函数（two-player minimax game）

$$\min_{\theta_G}\max_{\theta_D} V(G,D)=\sum_{c=1}^{V}(\mathbb{E}_{v\sim p_{true}(\cdot|v_c)}[\log D(v,v_c;\theta_D)]+(\mathbb{E}_{v\sim G(\cdot|v_c;\theta_G)}[\log(1-D(v,v_c;\theta_D))])$$

  - 给定θ_D，优化G：
    - 目标函数第一项和θ_G无关
    - 从生成器$G(v|v_c;\theta_G)$的角度来说，要提高判别器D预测G生成的样本v和 v_c之间存在边的概率值$D(v,v_c;\theta_D)$，也就是让$\left(1-D(v,v_c;\theta_D)\right)$小，因此整体是一个min的目标$min_{\theta_G}V(G,D)$;
    - 通过改变生成器θ_G继续生成节点。

算法原理

- 判别器$D(v, v_c; \theta_D)$的实现：
  - 非常简单的sigmoid 分类器

$$D(v, v_c) = \sigma(\mathbf{d}_v^\top \mathbf{d}_{v_c}) = \frac{1}{1 + \exp(-\mathbf{d}_v^\top \mathbf{d}_{v_c})}$$

  - 其中，d_v, d_{v_c} 是节点v和节点v_c在判别器D中的k维向量表达，因此 θ_D就可看作是所有 d_v的集合。
    - 它将节点v和节点v_c的embedding做内积，再用 sigmoid 函数进行处理，输出的是 edge 的概率 $p(edge(v, v_c)|v, v_c)$
    - 判别器也可以用其他判别式方法的模型，比如SDNE

算法原理

- 判别器$D(v, v_c; \theta_D)$的实现：

$$D(v, v_c) = \sigma(\mathbf{d}_v^\top \mathbf{d}_{v_c}) = \frac{1}{1 + \exp(-\mathbf{d}_v^\top \mathbf{d}_{v_c})}$$

  - 公式仅涉及v和v_c，因此，对于给定一个样本对(v, v_c)的情况下，只需要通过梯度上升法去更新对应的节点表达向量d_v和d_{v_c}：

$$\nabla_{\theta_D} V(G, D) = \begin{cases} \nabla_{\theta_D} \log D(v, v_c), \ if \ v \sim p_{\text{true}}; \\ \nabla_{\theta_D} \left(1 - \log D(v, v_c)\right), \ if \ v \sim G. \end{cases}$$

算法原理

- 生成器$G(v|v_c;\theta_G)$的实现：
  - 对于其他的所有节点，将G定义为softmax 函数

$$G(v|v_c) = \frac{\exp(\mathbf{g}_v^\top \mathbf{g}_{v_c})}{\sum_{v \neq v_c} \exp(\mathbf{g}_v^\top \mathbf{g}_{v_c})}$$

  - 其中$g_v, g_{v_c} \in \mathbb{R}^k$分别是生成器G的节点$v$和$v_c$的$k$ 维表示向量，而θ_G是所有g_v的并集。

  - 基于这样的设定，就可以按照上公式先计算出预估连接分布$G(v|v_c;\theta_G)$，根据这个概率分布对图进行随机采样可得到样本集合(v, v_c)。
  - 最后可以通过梯度下降法去优化更新θ_G。

算法原理

- 与常规GAN的生成过程中对连续随机变量z进行采样不同，GraphGAN 在为给定的v_c生成/预测未观察到的边(v, v_c)时，要求v是图中的真实节点。
- 由于节点v的采样是离散的，因此应用了强化学习中经常使用的策略梯度（policy gradient）计算$V(G,D)$相对于θ_G的梯度：

$$\nabla_{\theta_G} V(G,D) = \nabla_{\theta_G} \sum_{c=1}^{V} \mathbb{E}_{v\sim G(\cdot|v_c)}[\log(1 - D(v, v_c; \theta_D))]$$

$$= \sum_{c=1}^{V} \mathbb{E}_{v\sim G(\cdot|v_c)}[\nabla_{\theta_G} \log G(v|v_c) \log(1 - D(v, v_c))]$$

算法原理

- 通过策略梯度（policy gradient）计算$V(G,D)$相对于θ_G的梯度：

$$\nabla_{\theta_G} V(G,D) = \nabla_{\theta_G} \sum_{c=1}^{V} \mathbb{E}_{v\sim G(\cdot|v_c)} [\log(1 - D(v, v_c; \theta_D))]$$

$$= \sum_{c=1}^{V} \mathbb{E}_{v\sim G(\cdot|v_c)} [\nabla_{\theta_G} \log G(v|v_c) \log(1 - D(v, v_c))]$$

  - 梯度 $\nabla_{\theta_G} V(G,D)$其实可以看作是权重为 $log(1 -$

算法原理

- 通过简单的softmax去实现生成器中的预估连接分布，这样做会有一些缺点：
  - 传统的softmax，一般是要计算整个图中所有的节点的softmax值，这样的话，耗时会很大；
  - 网络自身的拓扑结构会隐含着丰富的信息，而softmax函数则完全忽略了这些，因为它是“公平”地看待每一个节点的，只是完成了归一化的任务；

算法原理

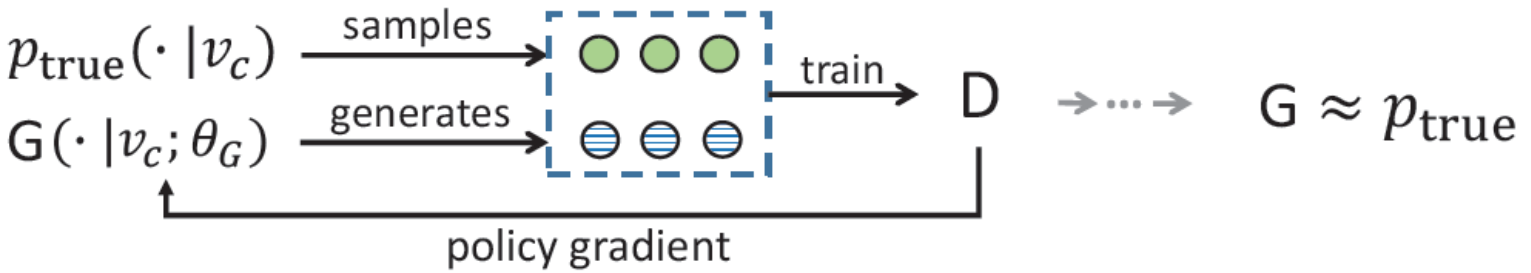
- Graph softmax

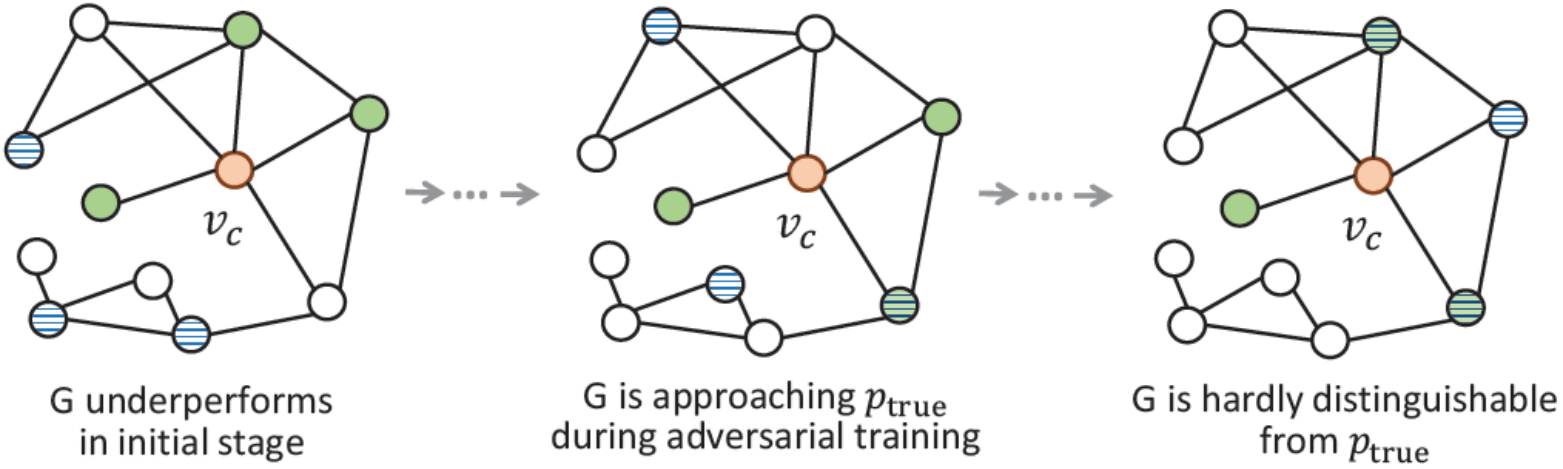
$$G(v|v_c) \triangleq \left(\prod_{j=1}^{m} p_c(v_{r_j}|v_{r_{j-1}})\right) \cdot p_c(v_{r_{m-1}}|v_{r_m})$$

  - 通过softmax嵌入网络结构信息，设计了一个基于宽度有限搜索的生成规则，使得生成器每次寻找邻居节点的时候不需要将v_c和图网络中的每一个其他节点进行计算
    - Normalized 归一化
    - Graph-structure-aware 感知图结构
    - Computationally efficient 计算效率高

算法原理

- 算法原理图





G underperforms in initial stage

G is approaching p_{true} during adversarial training

G is hardly distinguishable from p_{true}

算法原理

- 算法伪代码

Algorithm 2 GraphGAN framework

Require: dimension of embedding k, size of generating samples s, size of discriminating samples t
Ensure: generator $G(v|v_c;\theta_G)$, discriminator $D(v,v_c;\theta_D)$

```
Initialize and pre-train G(v|v_c;θ_G) and D(v,v_c;θ_D);
Construct BFS-tree T_c for all v_c ∈ V;
while GraphGAN not converge do
    for G-steps do
        G(v|v_c;θ_G) generates s vertices for each vertex v_c according to Algorithm 1;
        Update θ_G according to Eq. (4), (6) and (7);
    end for
    for D-steps do
        Sample t positive vertices from ground truth and t negative vertices from G(v|v_c;θ_G) for each vertex v_c;
        Update θ_D according to Eq. (2) and (3);
    end for
end while
return G(v|v_c;θ_G) and D(v,v_c;θ_D)
```

算法原理

Datasets

- arXiv-AstroPh: 18,772 vertices and 198,110 edges
- arXiv-GrQc: 5,242 vertices and 14,496 edges
- BlogCatalog: 10,312 vertices, 333,982 edges and 39 labels
- Wikipedia: 4,777 vertices, 184,812 edges and 40 labels
- MovieLens-1M: 6,040 users and 3,706 movies

Baselines

- DeepWalk (KDD 2014)
- LINE (WWW 2015)
- Node2vec (KDD 2016)
- Struc2vec (KDD 2017)

算法原理

- 算法执行结果
  - 链接预测任务

Table 1: Accuracy and Macro-F1 on arXiv-AstroPh and arXiv-GrQc in link prediction.

Model	arXiv-AstroPh		arXiv-GrQc	
	Acc	Macro-F1	Acc	Macro-F1
DeepWalk	0.841	0.839	0.803	0.812
LINE	0.820	0.814	0.764	0.761
Node2vec	0.845	0.854	0.844	0.842
Struc2vec	0.821	0.810	0.780	0.776
GraphGAN	**0.855**	**0.859**	**0.849**	**0.853**

算法原理

- 算法执行结果
  - 节点分类任务

Table 2: Accuracy and Macro-F1 on BlogCatalog and Wikipedia in node classification.

Model	BlogCatalog		Wikipedia	
	Acc	Macro-F1	Acc	Macro-F1
DeepWalk	0.225	0.214	0.194	0.183
LINE	0.205	0.192	0.175	0.164
Node2vec	0.215	0.206	0.191	0.179
Struc2vec	0.228	0.216	0.211	0.190
GraphGAN	**0.232**	**0.221**	**0.213**	**0.194**

优劣分析

- 横向对比
  - 用生成对抗的思想去更新网络节点向量表达
- 纵向对比
  - 类似的模型有ANE(Adversarial Network Embedding)（AAAI 2018）

应用总结

- 算法的应用领域
  - 节点分类
  - 链接预测
  - 社区发现
  - 推荐系统
- 未来的发展
  - 动态网络嵌入

参考文献

[1] Wang H , Wang J , Wang J , et al. GraphGAN: Graph Representation Learning with Generative Adversarial Nets[C]. AAAI 2018.
[2]《GraphGAN: Graph Representation Learning with GAN》阅读笔记 https://zhuanlan.zhihu.com/p/47622003

谢谢！

大成若缺，其用不弊。大盈若冲，其用不穷。大直若屈。大巧若拙。大辩若讷。静胜躁，寒胜热。清静为天下正。