GNN (Graph Neural Network)
Graph Embedding & GCN & GraphSAGE & GAT & DropEdge

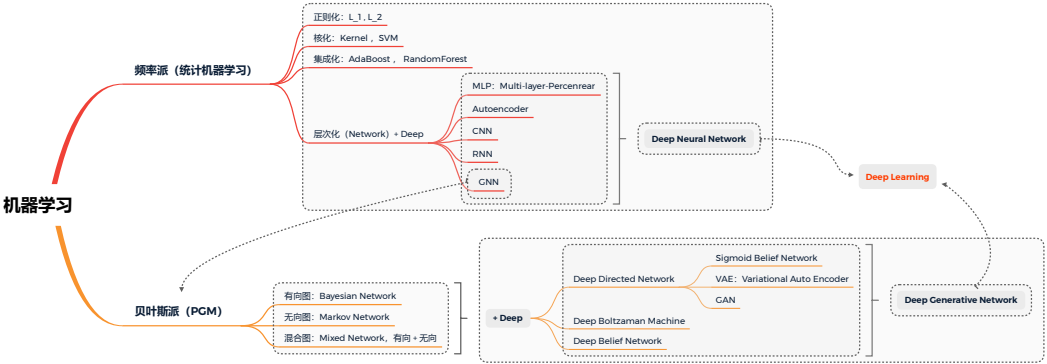
WISERCLUB – GNN

Xiamen University

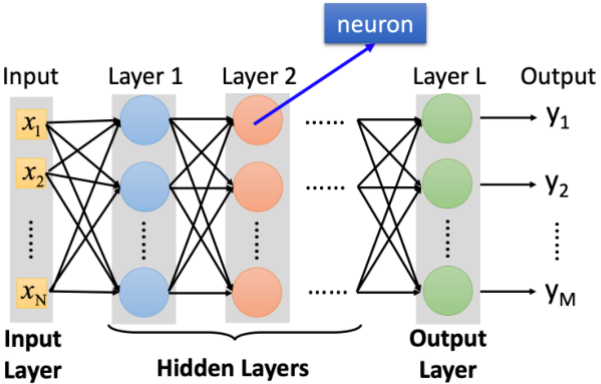
Dec, 2020

Outline

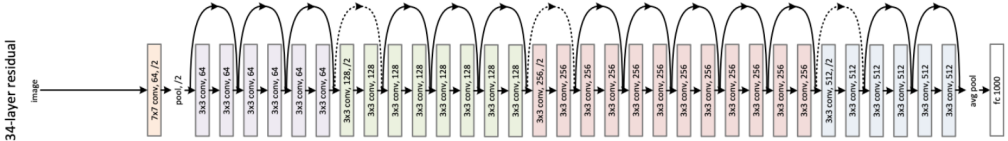
ML


机器学习
频率派（统计机器学习）
正则化：L_1，L_2
核化：Kernel，SVM
集成化：AdaBoost，RandomForest
层次化（Network）+ Deep
MLP：Multi-layer-Percenrear
Autoencoder
CNN
RNN
GNN
Deep Neural Network
Deep Learning
贝叶斯派（PGM）
有向图：Bayesian Network
无向图：Markov Network
混合图：Mixed Network，有向 + 无向
+ Deep
Deep Directed Network
Sigmoid Belief Network
VAE：Variational Auto Encoder
GAN
Deep Boltzaman Machine
Deep Belief Network
Deep Generative Network

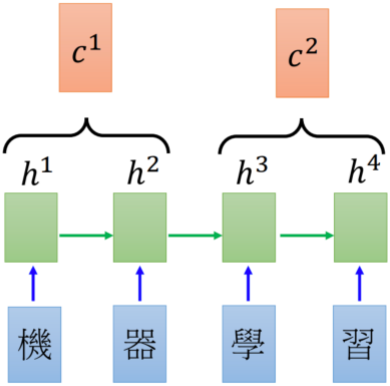
Neural Network

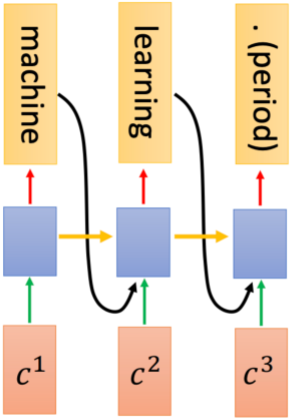


CNN

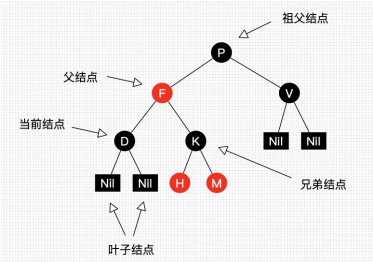

34-layer residual
image
7x7 conv, 64, /2
pool, /2
3x3 conv, 64
3x3 conv, 128, /2
3x3 conv, 128
3x3 conv, 256, /2
3x3 conv, 256
3x3 conv, 512, /2
3x3 conv, 512
avg pool
fc 1000

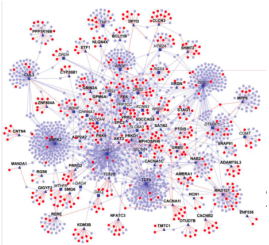
RNN

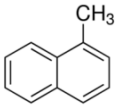

c^1
c^2
h^1
h^2
h^3
h^4
機
器
學
習
Encoder


machine
learning
. (period)
c^1
c^2
c^3
Decoder

A toy


祖父结点
父结点
当前结点
兄弟结点
叶子结点
P
F
V
D
K
Nil
Nil
Nil
Nil
H
M

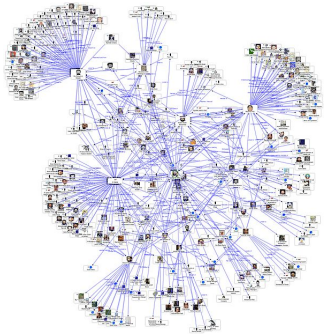



CH3


上海地铁线路图 Shanghai Metro Map

Graph I

- 社交网络，互联网，已连接的 IoT 设备，铁路网络或电信网络
- 网络是互连节点的集合。节点表示实体，它们之间的连接是某种关系。在图论中，这些网络称为图.



Social Network

Graph II

- 图数据库:
  - 社交网络
  - 实时推荐
  - 知识图谱
  - 反欺诈和风控
- 图模型:
  - 属性图（Property Graph）
  - 资源描述框架（RDF，Resource Description Framework）
  - 超图（HyperGraph）

GNN, Why ? I

平移不变性（Translation invariance）

在用基础的分类结构比如 ResNet、Inception 给一只猫分类时，无论猫怎么扭曲、平移，最终识别出来的都是猫，输入怎么变形输出都不变这就是平移不变性，网络的层次越深这个特性会越明显。

Euclidean domains data

最显著的特征是他们具有规则的空间结构，如图片是规则的正方形，语音是规则的一维序列等，这些特征都可以用一维或二维的矩阵来表示，可以共享参数。

- 1D：声音，时间序列等;
- 2D：图像等;
- 3D：视频，高光谱图像等

GNN, Why ? II

Remark

CNN 中的卷积就是一种离散卷积，本质上就是利用一个共享参数的过滤器（Kernel）。图数据和图像数据的差别在于节点邻居个数、次序都是不定的。

Non Euclidean data

生活中很多数据不具备规则的空间结构，称为 Non Euclidean data，如，推荐系统、电子交易、分子结构等抽象出来的图谱。流形也是典型的非欧结构。

- 1D：社交网络 (eg：Facebook，Twitter 等)
- 2D：生物网络 (基因，分子，大脑连接) 等
- 3D：基础设施网络 (eg：能源，交通，互联网，通信等)

GNN, Why ? III

社交网络 & 图数据

社交网络非常适合用图数据来表达，社交网络中节点以及节点与节点之间的关系，用户 A（有 ID 信息等）、用户 B、帖子都是节点，用户与用户之间的关系是关注，用户与帖子之间的关系可能是发布或者转发。

图数据中的空间特征

- 节点特征：每个节点有自己的特征；（体现在点上）
- 结构特征：图数据中的每个节点具有结构特征，即节点与节点存在一定的联系。（体现在边上）

GNN, Why ? IV

图卷积的核心思想

图卷积的核心思想是利用『边的信息』(Edge) 对『节点信息』(Vertex) 进行『聚合』(Aggregate) 从而生成新的『节点表示』。

- 节点分类—反欺诈：图中每个节点都拥有自己的特征信息, 通过该特征信息，我们可以构建一个风控系统，如果交易节点所关联的用户 IP 和收货地址与用户注册 IP 和注册地址不匹配，那么系统将有可能认为该用户存在欺诈风险。
- 边结构预测—商品推荐：图中每个节点都具有结构信息。如果用户频繁购买某种类别商品或对某种类别商品评分较高，那么系统就可以认定该用户对该类商品比较感兴趣，

Tasks, Dataset, and Benchmark

Tasks

- Semi-supervised node classification: Predict a type of a given node
- Regression
- Graph classification
- Graph representation learning: Users with the same label are located in closer
- Link prediction: Predict whether two nodes are linked

Dataset

- CORA: citation network. 2.7k nodes and 5.4k links
- TU-MUTAG: 188 molecules with 18 nodes on average

Definitions of graph I

graph

A graph G is a tuple $G = (V, E)$ of vertices V and edges E.

Degree

The degree of a vertex v of an undirected graph G = (V, E) is the number overtices that are connected to v by means of an edge,

Attributed graph

An attributed graph is a graph that has either labels or attributes on the nodes and/or edges.

Definitions of graph II

Walks

A sequence of k nodes $v_1, v_2, \cdots, v_k$ of the vertices of a graph G is called a walk of length k − 1 if the edge between two consecutive vertices exists.

Path

Vertices of a walk are allowed to repeat. If, however, node repetition is not allowed, one typically refers to the node sequence as a path,

Cycles

Adjective directed often being added in the case of directed graphs. Special consideration is given to cycles, i.e. walks of length k − 1 for which $v_1 = v_k$.

Definitions of graph III

Random walks

A walk in a graph is referred to as a random walk if the next vertex (or edge) is picked in a probabilistic manner.

k-hop neighbourhood of a vertex

Given a vertex v of a graph G and $k \in \mathbb{N}_{>0}$ its k-hop neighbourhood $N^{(k)}(v)$ is defined as all the vertices in G that are reachable in at most k steps, which includes v, assuming uniform edge weights.

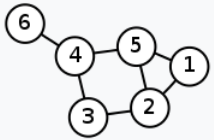
Adjacency matrix A

A graph with n vertices will thus be represented by an n $\times$ n binary matrix whose entry $A_{ij} = 1$ if the ith and jth vertex of the graph are connected by an edge.

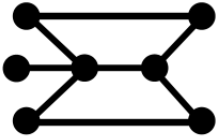
Definitions of graph IV

Graph Laplacian

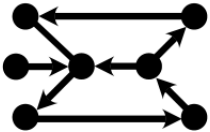
Let A be the adjacency matrix of a graph G = (V, E).let D be the degree matrix of G. This diagonal matrix contains the degree of each vertex $v \in V$ in the unweighted case. $L := D - A$

Labeled graph	Degree matrix	Adjacency matrix	Laplacian matrix
	$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$

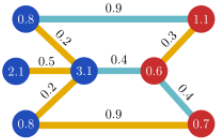
Definitions of graph V



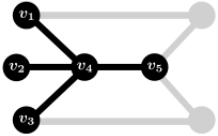
(a) An undirected graph



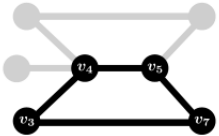
(b) A directed graph



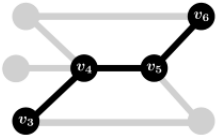
(c) An attributed graph



(d) $\mathrm{N}^{(1)}(v_4)$



(e) A cycle



(f) A shortest path

Laplacian Matrix

- 普通形式的拉普拉斯矩阵 $L = D - A$
- 对称归一化的拉普拉斯矩阵（Symmetric normalized Laplacian）

$$L^{\text{sys}} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} A D^{-1/2}$$

$$L_{i,j}^{sys} = \begin{cases} 1 & i = j \text{ and } \operatorname{diag}(v_i) \neq 0 \\ -\frac{1}{\sqrt{\operatorname{diag}(v_i)\operatorname{diag}(v_j)}} & i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

- 随机游走归一化拉普拉斯矩阵（Random walk normalized Laplacian）

$$L^{rw} = D^{-1} L = I - D^{-1} A$$

$$L_{i,j}^{rw} = \begin{cases} 1 & i = j \text{ and } \operatorname{diag}(v_i) \neq 0 \\ -\frac{1}{\operatorname{diag}(v_i)} & i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Definition

知乎：刘斯坦

- Embedding（嵌入）是拓扑学里面的词，在深度学习领域经常和 Manifold（流形）搭配使用。

Wikipedia

In mathematics, an embedding is one instance of some mathematical structure contained within another instance, such as a group that is a subgroup.

Tensorflow 社区

An embedding is a mapping from discrete objects, such as words, to vectors of real numbers.

A toy I

- User Data, 基础属性数据：王者荣耀游戏中玩家的性别、年龄、关系链、兴趣偏好 (射手，打野，皮肤）等
  - 对于用户兴趣偏好，一般简单地采用文本 embedding 方法来得到各标签的 embedding 向量，然后根据用户对个标签的偏好程度做向量加权；
  - 对于关系链数据（如同玩好友、相互关注等），构造用户关系图，然后采用基于图的 embedding 方法来得到用户的 embedding 向量；
- Item Data: 基本信息数据: 如标题、作者、游戏简介、标签等
  - 对于文本、简介和标签等可以采用基于文本的 embedding 方法来在已有语料上预训练模型，然后得到对应的 embedding 向量（如 word2vec 或者 BERT）；
  - 对于有明确关系的（如 item $\rightarrow$ 文本 $\rightarrow$ 标签 or 关键词）可以采用对关键词/标签的向量均值来表示 item 的文本向量（FaceBook 开源的**StarSpace**）；

A toy II

- User Behaviour Data: 用户在场景中的行为数据，如点击、互动、下载等
  - 针对用户对 Item 的操作（如点击、互动、下载）构造用户 → Item + Item Tag，构造用户-Item-Tag 的异构网络，然后可以采用 Metapath2vec 来得到各节点的 embedding 向量；
  - 记录用户在整个场景访问 Item，构造 Item-Item 关系图，然后采用 DeepWalk 得到 Item 的向量，用来挖掘 Item 间的关系特征；
- Extra Data: 外部扩充数据，如用户游戏行为、用户微信其他场景活跃等
  - 标签型（主要是用户在各场景的兴趣偏好）
  - 关系链型（如游戏中心好友、游戏内好友、开黑好友）可以采用用户关系构造用户关系图，采用 Graph embedding 方法（如 GraphSAGE）来表示用户抽象特征，Struct2vec

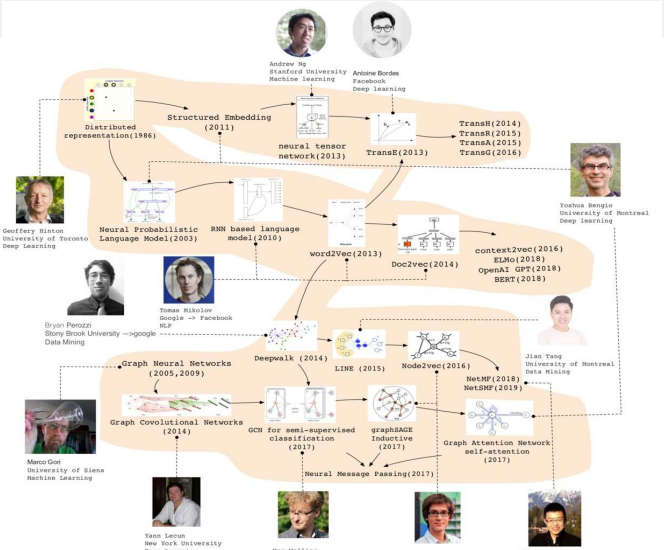
Embedding 分类 I

1. 经典矩阵分解法
   - Singular value decomposition $X_{m\times n} \approx U_{m\times r} S_{r\times r} U^T_{r\times n}$
   - Latent Factor Model(FunkSVD) $X_{m\times n} \approx P^T_{k\times m} Q_{k\times n}$
2. 基于内容的 Embedding 方法
   - 静态向量
     - Word2vec: 采用 CBOW 或 Skip-gram, 将词从 one-hot 编码向量映射成 d 维稠密向量:
     - GloVe: 无监督词向量学习, 基于全局词频统计（count-based & overall statistics)。
     - FastText: 文本分类模型（有监督学习）, 词向量则是 FastText 的一个副产物。
   - 动态变量
     - Embeddings from Language Models(ELMo)
     - Generative Pre-Training(GPT)
     - Bidirectional Encoder Representations from Transformers(BERT):
       Google 2018 年 10 月,NLP 领域“最靓的仔”

Embedding 分类 II

3. 基于 Graph 的 Embedding 方法
   - 浅层图模型 (Shallow)
     - DeepWalk：Random Walk + Word2Vec
     - LINE: explicitly preserves both **first-order** and **second-order** proximities.
     - PTE: learn **heterogeneous** text network embedding via a semisupervised manner.
     - Node2vec: use a **biased** random walk to better explore node's neighborhood.
     - Metapath2vec : **meta-path-based** random walks for heterogeneous networks.
     - GATNE: **inductive learning** for heterogeneous networks.
   - 深度图模型 (Deep)
     - 基于谱的 GCN (GCN)
     - 基于空间的 GCN (GraphSAGE)

GNN


Andrew Ng
Stanford University
Machine Learning
Antoine Bordes
facebook
Deep learning
Distributed representation(1986)
Structured Embedding (2011)
neural tensor network(2013)
TransE(2013)
TransH(2014)
TransR(2015)
TransA(2015)
TransG(2016)
Yoshua Bengio
University of Montreal
Deep learning
Geoffery Hinton
University of Toronto
Deep Learning
Neural Probabilistic Language Model(2003)
RNN based language model(2010)
word2Vec(2013)
Doc2vec(2014)
context2vec(2016)
ELMo(2018)
OpenAI GPT(2018)
BERT(2018)
Tomas Mikolov
Google -> Facebook
NLP
Bryan Perozzi
Stony Brook University ->google
Data Mining
Deepwalk (2014)
LINE (2015)
Node2vec(2016)
Jian Tang
University of Montreal
Data Mining
NetMF(2018)
NetNMF(2019)
Graph Neural Networks (2005,2009)
Graph Covolutional Networks (2014)
GCN for semi-supervised classification (2017)
graphSAGE Inductive (2017)
Graph Attention Network self-attention (2017)
Marco Gori
University of Siena
Machine Learning
Neural Message Passing(2017)
Yann Lecun
New York University
Deep Learning
Max Welling
University of Amsterdam
Statistical Learning
Jure Leskovec
Stanford University
Data Mining
Jie Tang
Tsinghua University
Data Mining

Deepwalk = RandomWalk + Word2Vec

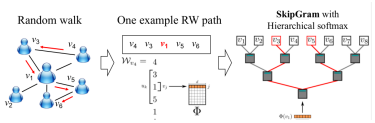
$$\Pr\left(v_i \mid (v_1, v_2, \cdots, v_{i-1})\right)$$

$$\Phi : v \in V \mapsto \mathbb{R}^{|V| \times d} \qquad P\left(\{v_{i-w}, \cdots, v_{i+w}\} \setminus v_i \mid \Phi\left(v_i\right)\right) = \prod_{j=i-w, j \neq i}^{i+w} P\left(v_j \mid \Phi\left(v_i\right)\right)$$

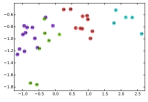
$$P\left(v_j \mid \Phi\left(v_i\right)\right) = \prod_{l=1}^{\log|V|} P\left(b_l \mid \Phi\left(v_i\right)\right) = \prod_{l=1}^{\log|V|} \frac{1}{1 + e^{-\Phi(v_i) \cdot \psi(b_l)}}$$

Remark

a mapping functio Φ represents the latent social representation associated with each vertex v in the graph, $b_{[\log|V|]} = \text{root}, b_l = u_k$.



(a) Input: Karate Graph



(b) Output: Representation

Deepwalk Algorithm

Algorithm 1 DeepWalk(G, w, d, γ, t)

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

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

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

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

Features:

- Randomly initialize the representations
- Each classifier in the hierarchy has a set of weights
- Use SGD (stochastic gradient descent) to update both classifier weights and vertex representations simultaneously

$$\mathcal{L} = \sum_{v \in V} \sum_{c \in W_v} -\log(P(c \mid v))$$

$$p(c \mid v) = \frac{\exp\left(\mathbf{z}_v^\top \mathbf{z}_c\right)}{\sum_{u \in V} \exp\left(\mathbf{z}_v^\top \mathbf{z}_u\right)}$$

Remark: DeepWalk (node representation learning) performs well, especially when labels are sparse! DeepWalk utilizes fixed-length, unbiased random walks to generate context for each node.

LINE:Definition

Definition of the first-order proximity

- The first-order proximity in a network is the local pairwise proximity between two vertices.
- For each pair of vertices linked by an edge (v_i, v_j) , w_{ij} is the weight of the edge: indicates the first-order proximity between v_i and v_j
- If no edge is observed between v_i and v_j , their first-order proximity is 0 .

Definition of the second-order proximity

- The second-order proximity between a pair of vertices (v_i, v_j) in a network is the similarity between their neighborhood network structures.
- If no vertex is linked from/to both v_i and v_j, the second-order proximity between v_i and v_j is 0 .

LINE:Information Network Embedding

Remark

- Given a large network $G = (V, E)$, LINE aims to represent each vertex $v \in V$ into a low-dimensional space $\mathbb{R}^d$
- Goal: learning a function $f_G : V \mapsto \mathbb{R}^{|d|}$, $|d| \ll |V|$,**both the first-order proximity and the second- order proximity between the vertices are preserved**
- For each node $v_i \in V$, we use $\overrightarrow{u}^i \in \mathbb{R}^{|d|}$ to represent the corresponding low-dimensional vector representation.

LINE with First-order Proximity

- For each undirected e_{ij}, we define the joint probability between vertex v_i and v_j as follows:

$$p_1(v_i, v_j) = \frac{1}{1 + \exp\left(-\vec{u'}_i^T \cdot \vec{u}_j\right)}$$

- It defines distribution $p(\cdot, \cdot)$ over the space $V \times V$, and its empirical probability can be defined as $\hat{p}_1(i, j) = \frac{w_{i,j}}{W}$ where $W = \sum_{i,j \in E} w_{i,j}$
- **Objective Function:** Minimize the following objective function, where $d(\cdot, \cdot)$ is the distance between two distributions.

$$O_1 = d\left(\hat{p_1}(\cdot, \cdot), p_1(\cdot, \cdot)\right)$$

- **Instantiate:** $d(\cdot, \cdot)$ as $\mathrm{KL} - \textit{divergence}$:

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

LINE with Second-order Proximity

- We first define the probability of "context" v_j generated by vertex v_i as:

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

- We minimize the following objective function:

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

- The empirical probability $\hat{p}_2(v_j \mid v_i)$ defined as:
  $\hat{p}_2(v_j \mid v_i) = \frac{w_{ij}}{d_j}$ where d_i is the out-degree of vertex i
- Replacing $d(\cdot, \cdot)$ with KL -divergence, setting $\lambda_i = d_i$, we have:

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

LINE Model Optimization

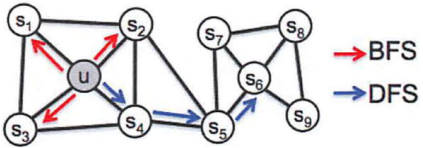
- Optimizing objective O_2 is computationally expensive. Why?
- Adopt the approach of **negative sampling** . More specifically, it specifies the following objective function for each edge $e_{i,j}$

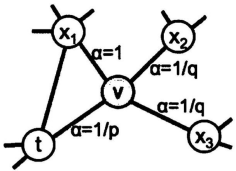
$$\underbrace{\log \sigma\left(\vec{u'}_j^T \cdot \vec{u}_i\right)}_{\text{the observed edges}} + \underbrace{\sum_{i=1}^{K} E_{v_n \sim P_n(v)}\left[\log \sigma\left(-\vec{u'}_n^T \cdot \vec{u}_i\right)\right]}_{\text{the negative edges drawn from the noise distribution}}$$

- We can use **asynchronous stochastic gradient algorithm(ASGD)** for optimizing above Eqn.

Node2vec: Biased Walks

- Use biased random walks to trade off **local and global** views of the network





- A flexible neighborhood sampling strategy : BFS? & DFS?
  - Breadth-first Sampling (BFS) :The neighborhood N_S is restricted to nodes which are immediate neighbors of the source. (**同质性**)
  - Depth-first Sampling (DFS) : The neighborhood consists of nodes sequentially sampled at increasing distances from the source node. (**结构性**)
- Biased walks is a special case of random walk,thus node2vec is a special case of DeepWalk,Why?

Node2vec: 同质性或结构性

- Biased random walk R that given a node v generates random walk neighborhood $N_{r,w}(v)$
- 通过控制节点间的跳转概率来控制 BFS 和 DFS 倾向性的,w_{vx} 是节点 v 和 x 边的权重.d_{tx} 这里表示节点与的最短路径，如 t 与 x_1 的最短路径为 1 (即 $\alpha(t, x_1)$)，

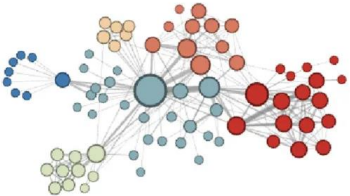
$$\pi_{vx} = \alpha_{p,q}(t, x) \cdot w_{vx}$$

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

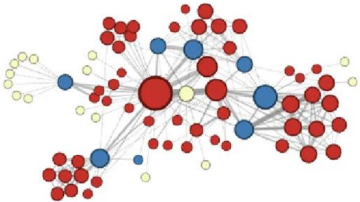
- Return parameter p : Return back to the previous node(BFS)
- In-out parameter q: Moving outwards(DFS)
- $p = q = 1$ Node2vec 退化成了 DeepWalk 算法

Node2vec: 优点 & 缺点

- 优点：可以对图中节点的**结构相似性和同质性**进行权衡，使模型更加灵活。



同质性



结构性

- 缺点：无法指定游走路径，且仅适用于解决只包含**一种类型节点**的同构网络，无法有效表示包含多种类型节点和边类型的复杂网络。$\Longrightarrow$ **Metapath2vec**

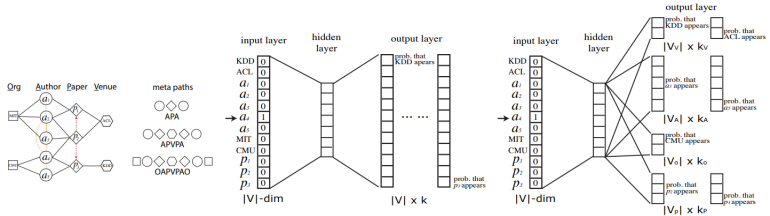
Metapath2vec: Heterogeneous Random Walk

- Input: a heterogeneous graph $G = (V, E, T)$

$G = (V, E, T)$

- A Heterogeneous Network is defined as a graph in which each node v and each link e are associate with their mapping functions $\phi(v) : V \to T_V$ and $\varphi(e) : E \to T_E$
- T_V, T_E denote the sets of object and relation types

- Output: $\mathbf{X} \in \mathbb{R}^{|V| \times k}$, $k \ll |V|$,k-dim vector $\mathbf{X}_v$ for each node v
- How do we random walk over **heterogeneous** networks?
- How do we apply skip-gram over **different types** of nodes?



(a) An academic network (b) Skip-gram in *metapath2vec*, node2vec, & DeepWalk (c) Skip-gram in *metapath2vec++*

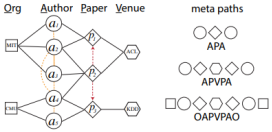
Why?

- Homogeneous Network Embedding: $p(c \mid v; \theta)$ defines the conditional probability of having a context node c given a node v.

$$\arg\max_{\theta} \prod_{v \in V} \prod_{c \in N(v)} p(c \mid v; \theta)$$

- Heterogeneous Skip-Gram: $p(c_t \mid v; \theta)$ is commonly defined as a softmax function.

$$\arg\max_{\theta} \sum_{v \in V} \sum_{t \in T_V} \sum_{c_t \in N_t(v)} \log p(c_t \mid v; \theta)$$



(a) An academic network

- Given a meta-path scheme

$$\mathcal{P}: V_1 \xrightarrow{R_1} V_2 \xrightarrow{R_2} \cdots V_t \xrightarrow{R_t} V_{t+1} \cdots \xrightarrow{R_{l-1}} V_l$$

- The transition probability at step i is defined as

$$p\left(v^{i+1} \mid v_t^i, \mathcal{P}\right) = \begin{cases} \frac{1}{\left|N_{t+1}\left(v_t^i\right)\right|} & \left(v^{i+1}, v_t^i\right) \in E, \phi\left(v^{i+1}\right) = t+1 \\ 0 & \left(v^{i+1}, v_t^i\right) \in E, \phi\left(v^{i+1}\right) \neq t+1 \\ 0 & \left(v^{i+1}, v_t^i\right) \notin E \end{cases}$$

- $v_t^i \in V_t$, N_{t+1} denote the V_{t+1} type of neighborhood of node v_t^i.
- Recursive guidance for random walkers, i.e.,

$$p\left(v^{i+1} \mid v_t^i\right) = p\left(v^{i+1} \mid v_1^i\right), \text{ if } t = l$$

- Heterogeneous negative sampling.

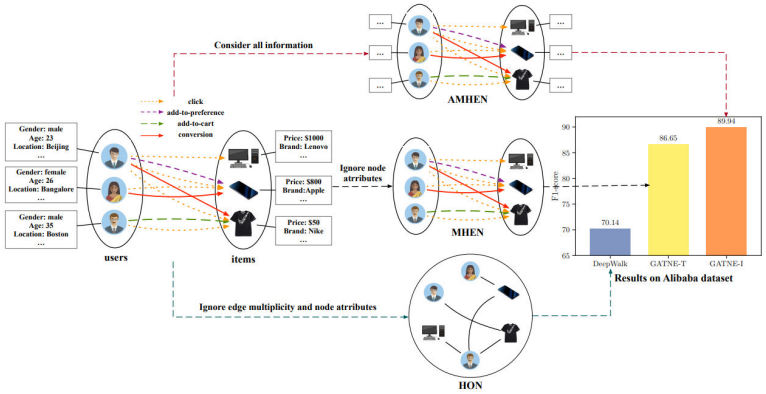
$$p\left(c_t \mid v; \theta\right)=\frac{e^{X_{c_t} \cdot X_v}}{\sum_{u_t \in V_t} e^{X_{u_t} \cdot X_v}} \tag{3}$$

$$\mathcal{O}(\mathbf{X})=\log \sigma\left(X_{c_t} \cdot X_v\right)+\sum_{k=1}^{K} \mathbb{E}_{u_t^k \sim P_t\left(u_t\right)}\left[\log \sigma\left(-X_{u_t^k} \cdot X_v\right)\right]$$

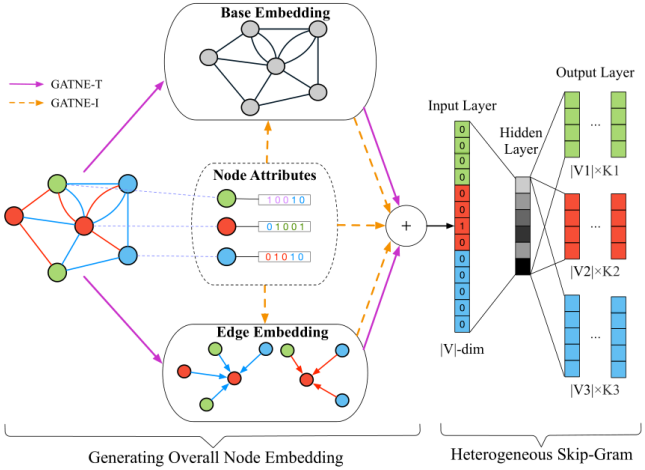
Remark: We further propose the metapath2vec++ framework, in which the softmax function is **normalized** with respect to **the node type of** the context c_t, Specifically, $p(c_t|v;\theta)$ is adjusted to the specific node type t .

GATNE: Attributed Multiplex Heterogeneous Network Embedding

- Recommend items to users by considering Attribute Multiplex Heterogeneous Networks (AMHEN)



Multiplex Heterogeneous Graph Embedding



Extension

- What are the fundamentals underlying the different models ?
- Can we unify the different graph embedding approaches ?

Skip gram with negative sampling(SGNS)

- SGNS maintains a multiset $\mathcal{D}$ that counts the occurrence of each word-context pair (w, c)
- Objective Function:

$$\mathcal{L} = \sum_{w} \sum_{c} \left(\#(w, c) \log g\left(x_w^T x_c\right) + \frac{b\#(w)\#(c)}{|\mathcal{D}|} \log g\left(-x_w^T x_c\right) \right)$$

- For sufficiently large dimension d, the objective above is equivalent to factorizing the PMI matrix. Why? Ans!

$$\log \frac{\#(w, c)|\mathcal{D}|}{b\#(w)\#(c)}$$

Reviewing DeepWalk algorithm

Algorithm 1: Reviewing random walk + skip gram

for $n = 1, 2, \ldots, N$ **do**

  Pick w_1^n according to a probability distribution $P(w_1)$. Generate a vertex sequence $(w_1^n, \cdots, w_L^n)$ of length L by a random walk on network G ;

  for $j = 1, 2, \ldots, L\text{-}T$ **do**

    for $r = 1, 2, \ldots, T$ **do**

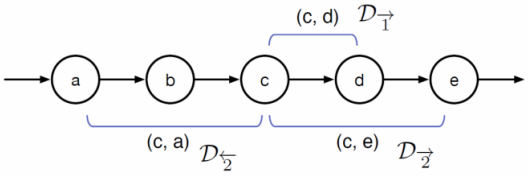
      Add vertex-context pair $\left(w_j^n, w_{j+r}^n\right)$ to multiset $\mathcal{D}$;

      Add vertex-context pair $\left(w_{j+r}^n, w_j^n\right)$ to multiset $\mathcal{D}$

Run SGNS on $\mathcal{D}$ with b negative samplpes.

Understanding DeepWalk

- How to interpret $\log\left(\frac{\#(w,c)|\mathcal{D}|}{b\#(w)\cdot\#(c)}\right)$ with graph structures?
- Partition the multiset $\mathcal{D}$ into several sub-multisets according to the way in which each node and its context appear in a random walk node sequence.



- More formally, for $r = 1, 2, \cdots, T$ .we define

$$\mathcal{D}_{\vec{r}} = \left\{(w,c) : (w,c) \in \mathcal{D}, w = w_j^n, c = w_{j+r}^n\right\}$$
$$\mathcal{D}_{\overleftarrow{r}} = \left\{(w,c) : (w,c) \in \mathcal{D}, w = w_{j+r}^n, c = w_j^n\right\}$$

Understanding random walk + skip gram I

$$\log\left(\frac{\#(w,c)|\mathcal{D}|}{b\#(w)\cdot\#(c)}\right)=\log\left(\frac{\frac{\#(w,c)}{|\mathcal{D}|}}{b^{\frac{\#(w)}{|\mathcal{D}|}\frac{\#(c)}{|\mathcal{D}|}}}\right) \quad \text{the length of random walk } L\to\infty$$

$$\frac{\#(w,c)}{|\mathcal{D}|}\xrightarrow{p}\frac{1}{2T}\sum_{r=1}^{T}\left(\frac{d_w}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{w,c}+\frac{d_c}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{c,w}\right) \quad \boldsymbol{P}=\boldsymbol{D}^{-1}\boldsymbol{A}$$

$$\frac{\#(w)}{|\mathcal{D}|}\xrightarrow{p}\frac{d_w}{\mathrm{vol}(G)} \qquad \frac{\#(c)}{|\mathcal{D}|}\xrightarrow{p}\frac{d_c}{\mathrm{vol}(G)}$$

$$\frac{\#(w,c)|\mathcal{D}|}{\#(w)\cdot\#(c)}=\frac{\frac{\#(w,c)}{|\mathcal{D}|}}{\frac{\#(w)}{|\mathcal{D}|}\cdot\frac{\#(c)}{|\mathcal{D}|}}\xrightarrow{p}\frac{\frac{1}{2T}\sum_{r=1}^{T}\left(\frac{d_w}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{\boldsymbol{w},\boldsymbol{c}}+\frac{d_c}{\mathrm{vol}(G)}\left(\boldsymbol{P}^r\right)_{\boldsymbol{c},\boldsymbol{w}}\right)}{\frac{d_w}{\mathrm{vol}(G)}\cdot\frac{d_c}{\mathrm{vol}(G)}}$$

$$=\frac{\mathrm{vol}(G)}{2T}\left(\frac{1}{d_c}\sum_{r=1}^{T}\left(\boldsymbol{P}^r\right)_{\boldsymbol{w},\boldsymbol{c}}+\frac{1}{d_w}\sum_{r=1}^{T}\left(\boldsymbol{P}^r\right)_{c,\boldsymbol{w}}\right)$$

Understanding random walk + skip gram II

$$\frac{\#(w,c)|\mathcal{D}|}{\#(w)\cdot\#(c)} \xrightarrow{p} \frac{\mathrm{vol}(G)}{2T}\left(\frac{1}{d_c}\sum_{r=1}^{T}\left(\boldsymbol{P}^r\right)_{w,c}+\frac{1}{d_w}\sum_{r=1}^{T}\left(\boldsymbol{P}^r\right)_{c,w}\right)$$

$$\frac{\mathrm{vol}(G)}{2T}\left(\sum_{r=1}^{T}\boldsymbol{P}^r\boldsymbol{D}^{-1}+\sum_{r=1}^{T}\boldsymbol{D}^{-1}\left(\boldsymbol{P}^r\right)^{\top}\right)$$

$$=\frac{\mathrm{vol}(G)}{2T}\Big(\sum_{r=1}^{T}\underbrace{D^{-1}\boldsymbol{A}\times\cdots\times D^{-1}A}_{r\text{ terms}}\boldsymbol{D}^{-1}+\sum_{r=1}^{T}\boldsymbol{D}^{-1}\underbrace{AD^{-1}\times\cdots\times AD^{-1}}_{r\text{ terms}}\Big)$$

$$=\frac{\mathrm{vol}(G)}{T}\sum_{r=1}^{T}\underbrace{\boldsymbol{D}^{-1}\boldsymbol{A}\times\cdots\times D^{-1}\boldsymbol{A}}_{r\text{ terms}}D^{-1}=\mathrm{vol}(G)\left(\frac{1}{T}\sum_{r=1}^{T}\boldsymbol{P}^r\right)\boldsymbol{D}^{-1}$$

DeepWalk is factorizing a matrix

- DeepWalk is asymptotically and implicitly factorizing

$$\log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(D^{-1}A\right)^{r}\right)D^{-1}\right) \qquad \mathrm{vol}(G)=\sum_{i}\sum_{j}A_{ij}$$

Remark

A : Adjacency matrix b : # negative samples
D : Degree matrix T : context window size

Extension of LINE

- Objective Function of LINE

$$\mathcal{L} = \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} \left(\boldsymbol{A}_{i,j} \log g\left(\boldsymbol{x}_i^\top \boldsymbol{y}_j \right) + \frac{b d_i d_j}{\mathrm{vol}(G)} \log g\left(-\boldsymbol{x}_i^\top \boldsymbol{y}_j \right) \right)$$

- Align it with the Objective of SNGS

$$\mathcal{L} = \sum_{w} \sum_{c} \left(\#(w, c) \log g\left(\boldsymbol{x}_w^\top \boldsymbol{y}_c \right) + \frac{b \#(w) \#(c)}{|\mathcal{D}|} \log g\left(-\boldsymbol{x}_w^\top \boldsymbol{y}_c \right) \right)$$

- LINE is actually factorizing:

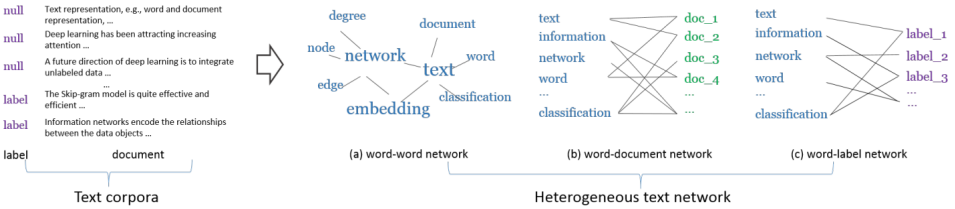
$$\log \left(\frac{\mathrm{vol}(G)}{b} \boldsymbol{D}^{-1} \boldsymbol{A} \boldsymbol{D}^{-1} \right)$$

- Recall DeepWalk's matrix form.When $T = 1$, LINE is the Special case of DeepWalk.

$$\log \left(\frac{\mathrm{vol}(G)}{b} \left(\frac{1}{T} \sum_{r=1}^{T} \left(\boldsymbol{D}^{-1} \boldsymbol{A} \right)^r \right) \boldsymbol{D}^{-1} \right)$$

PTE: Predictive Text Embedding about Heterogeneous Text Networks

$$\log\left(\begin{bmatrix} \alpha\,\mathrm{vol}\left(G_{\mathrm{ww}}\right)\left(\boldsymbol{D}_{\mathrm{row}}^{\mathrm{ww}}\right)^{-1}\boldsymbol{A}_{\mathrm{ww}}\left(\boldsymbol{D}_{\mathrm{col}}^{\mathrm{ww}}\right)^{-1} \\ \beta\,\mathrm{vol}\left(G_{\mathrm{dw}}\right)\left(\boldsymbol{D}_{\mathrm{row}}^{\mathrm{dw}}\right)^{-1}\boldsymbol{A}_{\mathrm{dw}}\left(\boldsymbol{D}_{\mathrm{col}}^{\mathrm{dw}}\right)^{-1} \\ \gamma\,\mathrm{vol}\left(G_{\mathrm{lw}}\right)\left(\boldsymbol{D}_{\mathrm{row}}^{\mathrm{lw}}\right)^{-1}\boldsymbol{A}_{\mathrm{lw}}\left(\boldsymbol{D}_{\mathrm{col}}^{\mathrm{lo}}\right)^{-1} \end{bmatrix}\right) - \log b$$



Extension:Node2vec

- Theorem node2vec is asymptotically and implicitly factorizing a matrix whose entry at $w-th$ row, $c-th$ column is

$$\log\left(\frac{\frac{1}{2T}\sum_{r=1}^{T}\left(\sum_u \boldsymbol{X}_{w,u}\underline{\boldsymbol{P}}^r_{c,w,u}+\sum_u \boldsymbol{X}_{c,u}\underline{\boldsymbol{P}}^r_{w,c,u}\right)}{b\left(\sum_u \boldsymbol{X}_{w,u}\right)\left(\sum_u \boldsymbol{X}_{c,u}\right)}\right)$$

- Read paper!
- Q: Can we directly factorize the derived matrices for learning embeddings?

Summary: Extension of Matrix I

DeepWalk $\log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$

LINE $\log\left(\frac{\mathrm{vol}(G)}{b}\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{D}^{-1}\right)$

PTE $\log\left(\begin{bmatrix}\alpha\,\mathrm{vol}\left(G_{\mathrm{ww}}\right)\left(\boldsymbol{D}_{\mathrm{row}}^{\mathrm{ww}}\right)^{-1}\boldsymbol{A}_{\mathrm{ww}}\left(\boldsymbol{D}_{\mathrm{col}}^{\mathrm{ww}}\right)^{-1}\\ \beta\,\mathrm{vol}\left(G_{\mathrm{dw}}\right)\left(\boldsymbol{D}_{\mathrm{row}}^{\mathrm{dw}}\right)^{-1}\boldsymbol{A}_{\mathrm{dw}}\left(\boldsymbol{D}_{\mathrm{col}}^{\mathrm{dw}}\right)^{-1}\\ \gamma\,\mathrm{vol}\left(G_{\mathrm{lw}}\right)\left(\boldsymbol{D}_{\mathrm{row}}^{\mathrm{lw}}\right)^{-1}\boldsymbol{A}_{\mathrm{lw}}\left(\boldsymbol{D}_{\mathrm{col}}^{\mathrm{lw}}\right)^{-1}\end{bmatrix}\right)-\log b$

Node2vec $\log\left(\frac{\frac{1}{2T}\sum_{r=1}^{T}\left(\sum_{u}\boldsymbol{X}_{w,u}\underline{\boldsymbol{P}}_{c,w,u}^{r}+\sum_{u}\boldsymbol{X}_{c,u}\underline{\boldsymbol{P}}_{w,c,u}^{r}\right)}{b\left(\sum_{u}\boldsymbol{X}_{w,u}\right)\left(\sum_{u}\boldsymbol{X}_{c,u}\right)}\right)$

Summary: Extension of Matrix II

How to choose Embedding

- 如果你的问题更加关注内容相似性 (局部邻域相似性) 那么你可以选择 node2vec, LINE, GraRep 等;
- 如果你的问题更加关注结构相似性，你可以选择 struc2vec,
- 如果你想处理大规模易变图，你可以采用 GraphSAGE, 或者先使用其他 GE 方法，再使用 GraphSAGE 归纳学习;
- 如果你想微调你的模型，你可选择 GraphGAN;
- Anyway, 选用什么模型取决你的问题!

Remark of struc2vec : 在风控领域，你可信并不能代表你的邻居可信 (有些“大 V”节点邻居众多)，但是一个直观的感觉是，如果两个人在图中处于相似的地位 (比如两个“大 V")，那么这两个人应该都可信或都不可信，并且-般来说这样的两个人 (节点) 都相聚比较远。

NetMF

- DeepWalk (random walk + skip-gram) is implicitly factorizing

$$\boldsymbol{M} = \log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$$

- Remark: Recall $\boldsymbol{z}_v^T\boldsymbol{z}_c$ the probability that node v and context c appear on a random walk path
- NetMF is explicitly factorizing

$$\boldsymbol{M} = \log\left(\frac{\mathrm{vol}(G)}{b}\left(\frac{1}{T}\sum_{r=1}^{T}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{r}\right)\boldsymbol{D}^{-1}\right)$$

- Remark: $\boldsymbol{z}_v^T\boldsymbol{z}_c$ the similarity score $\boldsymbol{M}_{v,c}$ between node v and context c defined by this matrix

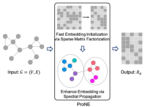
Challenge in NetMF

- NetMF is always a dense matrix.
- How to Solve? DO IT BY YOURSELF!

Graph Embedding: Why from Shallow to Deep

- Limitations of shallow encoding:
  - $O(|V|)$ parameters are needed:here no parameter sharing and every node has its own unique embedding vector
  - Inherently "transductive" : It is impossible to generate embeddings for nodes that were not seen during training
  - Do not incorporate node features: Many graphs have features that we can and should leverage.
- We will discuss deeper methods based on graph neural networks,i.e. Encoder is complex function that depends on graph structure.

A Quick Summary



Cognitive Graph — 2019: Ding et al., ACL'19

ProNE, GATNE, BurstGraph — 2019: Zhang et al., IJCAI'19; Cen et al., KDD'19; Zhao et al., IJCAI'19

FastGCNs, Graph attention — 2018: Velickovic et al., ICLR'18, Chen et al., ICLR 2018

NetMF & NetSMF — 2018: Qiu et al., WSDM'18 & WWW'19

Neural message passing, GraphSage — 2017: Gilmer et al., ICML'17; Hamilton et al., NIPS'17

Gated graph neural network — 2016: Li et al., ICLR'16

structure2vec — 2016: Dai et al., ICML'16

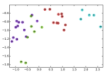
Graph convolutional network — 2015: Duvenaud et al., NIPS'15; Kipf & Welling ICLR'17

PTE, metapath2vec — 2015: Tang et al., KDD'15; Dong et al., KDD'17

LINE, node2vec — 2015: Tang et al., WWW'15; Grover & Leskovec, KDD'16

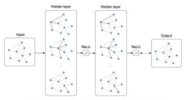
DeepWalk — 2014: Perozzi et al., KDD'14

Spectral graph convolution — 2014: Bruna et al., ICLR'14



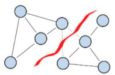
word2vec (skip-gram) — 2013: Mikolov et al., ICLR'13

Graph neural network — 2005: Gori et al., IJCNN'05



Spectral clustering — 2000: Ng et al. & Shi, Malik

Spectral partitioning — 1973: Donath & Hoffman



万物皆可 Embedding

1. 案例：GraphSAGE+FM+Transformer 强强联手：评微信的 GraphTR 模型
2. Graph Embedding 之如何使用 DeepWalk 从图中提取特征
3. Graph Neural Network（GNN）综述
4. 用万字长文聊一聊 Embedding 技术
5. 深度学习推荐系统中各类流行的 Embedding 方法
6. 一文梳理推荐系统中 Embedding 应用实践
7. 推荐系统中稀疏特征 Embedding 的优化表示方法
8. 异质图嵌入综述: 方法、技术、应用和资源, 23 页 pdf
9. 清华大学朱文武老师最新「图表示深度学习」的 5 种方法综述论文，51 页 pdf
10. Department of Computer Science,Tsinghua University, 唐杰

GNN 分类

- GCN : Graph Convolution Networks
  - 基于空间: spatial-based
  - 基于谱: spectral-based
- GAT : Graph Attention Networks
- GA: Graph Autoencoders $\Longrightarrow$ SDNE
- GGN : Graph Generative Networks $\Longrightarrow$ GraphGAN
- GSN : Graph Spatial-temporal Networks $\Longrightarrow$ Diffusion-convolutional neural networks(DCNN)

GNN Roadmap–Convolution

Spatial-Based

Aggregation	Method
Sum	NN4G
Mean	DCNN & DGC & GraphSAGE
Weighted sum	MoNET & GAT & GIN
LSTM	GraphSAGE
Max Pooling	GraphSAGE

Spectral-Based

ChebNet $\Longrightarrow$ GCN $\Longrightarrow$ HyperGCN

拉普拉斯算子

定义

$$\Delta f = \nabla^2 f = \nabla \cdot \nabla f = div(gradf)$$

离散形式的拉普拉斯算子

$$\frac{\partial f}{\partial x} = f'(x) = f(x+1) - f(x)$$

$$\frac{\partial^2 f}{\partial x^2} = f''(x) \approx f'(x) - f'(x-1) = f(x+1) + f(x-1) - 2f(x) \Longrightarrow$$

$$\begin{aligned}\Delta f =& \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \\ =& f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)\end{aligned}$$

拉普拉斯矩阵

- 谱聚类中的拉普拉斯矩阵可以理解为是对图的一种矩阵表示形式。

$$
\begin{aligned}
\Delta f &= \begin{pmatrix} \Delta f_1 \\ \vdots \\ \Delta f_N \end{pmatrix} = \begin{pmatrix} d_1 f_1 - w_{1:} f \\ \vdots \\ d_N f_N - w_{N:} f \end{pmatrix} \\
&= \begin{pmatrix} d_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & d_N \end{pmatrix} f - \begin{pmatrix} w_{1:} \\ \vdots \\ w_{N:} \end{pmatrix} f \\
&= \operatorname{diag}(d_i) f - W f \\
&= (D - W) f \\
&= L f
\end{aligned}
$$

GCN-Spectral Based I

特征函数与傅里叶算子:

$$F(w) = \mathcal{F}[f(t)] = \int_t f(t)e^{-iwt}\mathrm{dt}$$

$$e^{-iwt} = cos(wt) - isin(wt)$$

$$\Delta e^{-iwt} = \frac{\partial^2 e^{-iwt}}{\partial t^2} = \frac{\partial - iwe^{-iwt}}{\partial t} = i^2 w^2 e^{-iwt} = -w^2 e^{-iwt}$$

GCN-Spectral Based II

$$\hat{f} = u^T x \Longrightarrow \hat{y} = g_\theta \star x = g_\theta(\Lambda)\hat{x} = g_\theta(\Lambda)u^T x \Longrightarrow$$

$$y = \left(u^T\right)^{-1}\hat{y} = u g_\theta(\Lambda)\hat{x} = g_\theta(u\Lambda u^T)x = g_\theta(L)x$$

$$L_{reg} = \sum_i \sum_j W_{ij}\|f(X_i) - f(X_j)\| = f^T(X)Lf(X)$$

$$L = I_n - D^{-\frac{1}{2}}WD^{-\frac{1}{2}} = u\Lambda u^T$$

Remark

$L_{n\times n}$ 在无向图中是一个实对称阵，实对称阵可对正交对角化，n 个不同的特征向量可以表示 n 个基向量（Independent）。

GCN-Spectral Based III

Cont

$u^T_{n\times n} f_{n\times 1}$ 表示将 f 所在的空域转换为时域，g_θ 表示一个关于 θ 的 $n \times n$ 的对角矩阵，其中的参数也就是机器学习要学习的参数。$\Lambda_{n\times n}$ 为 L 的正交对角化矩阵，$uu^T = I$。

$$\hat{x} = \begin{pmatrix} \hat{f}(\lambda_1) \\ \hat{f}(\lambda_2) \\ \vdots \\ \hat{f}(\lambda_n) \end{pmatrix} = \begin{pmatrix} u_1(1) & u_1(1) & \cdots & u_1(n) \\ u_2(1) & u_2(2) & \cdots & u_2(n) \\ \vdots & \vdots & \vdots & \vdots \\ u_n(1) & u_n(2) & \cdots & u_n(n) \end{pmatrix} \begin{pmatrix} f(1) \\ f(2) \\ \vdots \\ f(n) \end{pmatrix}$$

GCN-Spectral Based IV

$$\hat{f}(\lambda_1) = u_1(1) f(1) + u_1(2) f(2) + \cdots + u_1(n) f(n) = \sum_{i=1}^{n} u_1(i) x(i) \Longrightarrow$$

$$f(\hat{\lambda}_\ell) = \sum_{i=1}^{n} u_\ell(i) f(i) = \int u_\ell(i) f(i) \mathrm{di} \quad \ell = 1, 2, \ldots, n$$

GCN-Spectral Based V

傅里叶逆变换

$$f(t) = \mathcal{F}^{-1}(F(w)) = \frac{1}{2\pi}\int F(w)e^{iwt}\mathrm{dw}$$

$$f \star h = \mathcal{F}^{-1}\left(\hat{f(w)} * \hat{h(w)}\right) = \frac{1}{2\pi}\int \left(\hat{f(w)} * \hat{h(w)}\right)\mathrm{dw}$$

$$(f \star h)_G = u\, diag\left[\hat{h(w)}\right] u^T f = u\, diag\left[u^T h\right] u^T f = u\left[(u^T h) \odot (u^T f)\right]$$

Remark

$\odot$ 表示 Hadamard product（哈达马积），对于两个维度相同的向量、矩阵、张量进行对应位置的逐元素乘积运算。

GCN-Spectral Based VI

Remark

h 为卷积核,$\hat{h}$ 为 h 的傅里叶变化，这里的 $diag\left[h(\hat{w})\right]$ 是上文中所说的 $g_\theta(\Lambda)$ 的另一种形式，λ 在频域中和频率表示一个含义。

$$diag(\hat{h}) = \begin{pmatrix} h(\hat{\lambda}_1) & 0 & 0 & 0 \\ 0 & h(\hat{\lambda}_2) & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & h(\hat{\lambda}_n) \end{pmatrix}$$

如何用特征值视角理解总变差?

- 图信号的总变差与图的特征值之间有着非常直接的线性对应关系，总变差是图的所有特征值的一个线性组合，权重是图信号相对应的傅里叶系数的平方。
  - 将特征值等价成频率，特征值越低，频率越低，对应的傅里叶基就变化得越缓慢，相近节点上的信号值趋于一致
  - 特征值越高，频率越高，对应的傅里叶基就变化得越剧烈，相近节点上的信号值非常不一致
- 结合傅里叶变换的本质理解，其实就是将图信号转换为了不同维度的频率解释

第一代 GCN

$$y_{output} = \sigma\left(U g_\theta(\Lambda) U^T x\right)$$

$$g_\theta(\Lambda) = \begin{pmatrix} \theta_1 & & & \\ & \theta_2 & & \\ & & \cdots & \\ & & & \theta_n \end{pmatrix}$$

局限性：

- 卷积核需要 n 个参数，参数数目极其地多
- 每一次前向传播都要计算矩阵的乘积。这对于大规模的 graph，运算量极其大（n 平方）
- 不具有局部连接性

第二代 GCN

$$y_{\text{output}} = \sigma\left(U g_\theta(\Lambda) U^T x\right)$$

$$g_\theta(\Lambda) = \begin{pmatrix} \sum_{j=0}^{K-1} \alpha_j \lambda_1^j & & & \\ & \sum_{j=0}^{K-1} \alpha_j \lambda_2^j & & \\ & & \cdots & \\ & & & \sum_{j=0}^{K-1} \alpha_j \lambda_n^j \end{pmatrix} = \sum_{j=0}^{K-1} \alpha_j \Lambda^j$$

局限性：

- j 是幂运算，不是标号
- $L^2 = U\Lambda U^T U\Lambda U^T = U\Lambda^2 U^T \Longrightarrow$
  $U\sum_{j=0}^{K-1} \alpha_j \Lambda^j U^T = \sum_{j=0}^{K-1} \alpha_j U\Lambda^j U^T = \sum_{j=0}^{K-1} \alpha_j L^j$

第二代 GCN

卷积层计算式就变为:

$$y_{\text{output}} = \sigma\left(\sum_{j=0}^{K-1} \alpha_j L^j x\right)$$

第二代 GCN 的优点

1. 只有 K 个参数，一般来说 K 是远小于节点数 n，相比于第一代 GCN 有大提升
2. 经过矩阵变换后，不需要做特征分解，只需要用拉普拉斯矩阵直接参与运算。但是注意运算复杂度没有发生变化，因为我们要计算拉普拉斯矩阵的幂
3. 具有局部连接性

第二代 GCN

- 局部连接性: 其实每次卷积就是将中心结点的 K 阶邻居上的图信号进行加权求和，权系数为 α
- 用之前第一代 GCN 同样的例子进行卷积运算，会发现大于 K 阶邻居的节点运算后为 0，而不是全都非 0

Chebnet

Why Chebnet?

引入切比雪夫多项式对卷积核进行逼近，利用 2 阶情况下的切比雪夫多项式，我们会得到一个线性运算，这将对我们进行深层网络构建提供基础。

- 第一类切比雪夫多项式:
  - $T_0(x) = 1$
  - $T_1(x) = x$

    $T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$
- Review of GCN-2th
  - $y_{\text{output}} = \sigma\left(Ug_\theta(\Lambda)U^T x\right)$

    $g_\theta(\Lambda) = \sum_{k=0}^{K-1} \theta_k T_k(\tilde{\Lambda})$

Chebnet

$$\tilde{\Lambda} = \frac{2}{\lambda_{\max}}\Lambda - I_N$$

$$g_\theta \star x = U\sum_{k=0}^{K-1}\theta_k T_k(\tilde{\Lambda})U^T x = \sum_{k=0}^{K-1}\theta_k U T_k(\tilde{\Lambda})U^T x = \sum_{k=0}^{K-1}\theta_k T_k(\tilde{L})x$$

$$\tilde{L} = \frac{2}{\lambda_{\max}}L - I_N$$

Remark

- 拉普拉斯矩阵是半正定矩阵，特征值不会为负
- 对角阵除以最大特征值归一化到 [0,1] 上
- 线性形式的切比雪夫多项式，即 K=2。
- 可以证明对称规范化的拉普拉斯矩阵的特征值是小于等于 2

Chebnet

- 为了简化运算我们把特征值最大值设为 2:

$$T_0(L) = I_N \quad T_1(\tilde{L}) = \tilde{L} = \frac{2}{\lambda_{\max}} L - I_N = L - I_N$$

- 卷积运算式简化为：

$$g_\theta \star x = \theta_0 x + \theta_1 (L - I_N) x = \theta_0 x - \theta_1 D^{-\frac{1}{2}} A D^{-\frac{1}{2}} x$$

- 为了简化运算 *Suppose* $\theta = \theta_0 = -\theta_1$

$$g_\theta \star x = \theta \left(I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \right) x$$

- 特征值范围为 [0,2]，这意味着层数很深的时候可能会引起梯度爆炸或消失，所以要再次规范化。

$$I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$$
$$\tilde{A} = A + I_N$$
$$\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$$

Chebnet

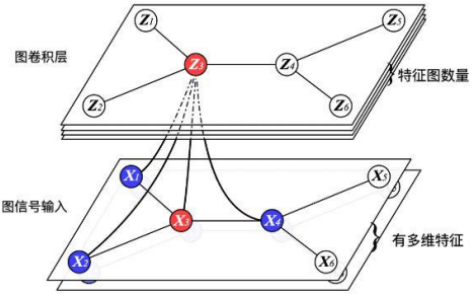
$$Z = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X\Theta$$
$$Z = f(X, A) = \text{softmax}\left(\hat{A}\,\text{RELU}\left(\hat{A}X\Theta^{(1)}\right)\Theta^{(2)}\right)$$
$$\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$$

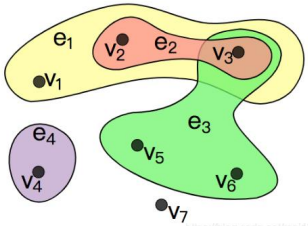
Remark

- softmax 函数将里面得到的数值映射到 [0,1], 便于模型做出预测
- H 为每一层的输入和输出。W 是线性变换，也就是上式中的 Θ

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\,\widetilde{D}^{-\frac{1}{2}}\,H^{(l)}W^{(l)}\right)$$

Chebnet & HyperGCN





- 超图：普通的图结构就是一条边只能连接 2 个节点，但是在超图之中，一条边可以连接多个节点。
- 对于一个超边，我们选取其中包含节点隐含的信息的二阶范数的最大的两个节点为带权简单图组成的边。

GCN 的优点：

- 局部连接
- 参数共享
- End to End 学习

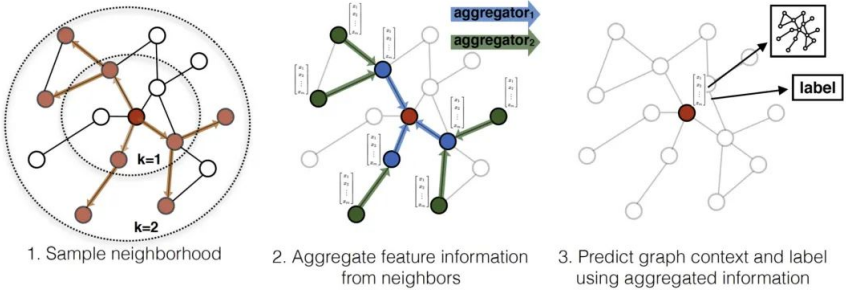
GCN 的缺陷：

- 如果图结构发生变化，一切推翻重来
- 产生过平滑的问题

Remark: 随着层数的增多，L 的 k 次方里高阶邻居的系数会越来越大，低阶邻居的系数会越来越小，然后两者逐渐趋于一致。

GraphSAGE

- 直推式学习：直接根据 graph 学习到个 *NxF* 的矩阵，graph 结构变化是要重新学习的
- 归纳式学习：利用节点邻域信息直接学习出新增节点 (unseen node) 的 embedding 表示



1. Sample neighborhood

2. Aggregate feature information from neighbors

3. Predict graph context and label using aggregated information

GraphSAGE

- GCN 的局限：无法进行 Inductive Learning
  - 卷积核依赖于整体结构决定的拉普拉斯特征基，加入新节点会改变图的结构，因而只能预测子图，无法直接预测新节点。
- GraphSAGE：解决了 GCN 不能 inductive learning 的问题，相较 GCN 更加灵活，能以较小代价预测新节点
- 主要思想：同时训练单个节点的 Embedding 与聚合函数

Transductive Learning VS. Inductive Learning

Transductive learning 与 Inductive Learning 的区别：

- 模型训练：Transductive learning 在训练过程中已经用到测试集数据（不带标签）中的信息，而 Inductive learning 仅仅只用到训练集中数据的信息。
- 模型预测：Transductive learning 只能预测在其训练过程中所用到的样本（Specific $\rightarrow$ Specific）。而 Inductive learning，只要样本特征属于同样的欧拉空间，即可进行预测（Specific $\rightarrow$ Gerneral）
- 模型复用性：当有新样本时，Transductive learning 需要重新进行训练；Inductive Leaning 则不需要。
- 模型计算量：Transductive Leaning 需要更大的计算量，即使其有时候确实能够取得相比 Inductive learning 更好的效果。

GraphSAGE——前向传播

Algorithm 1: GraphSAGE embedding generation (i.e., forward propagation) algorithm

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; input features $\{\mathbf{x}_v, \forall v \in \mathcal{V}\}$; depth K; weight matrices $\mathbf{W}^k, \forall k \in \{1, ..., K\}$; non-linearity σ; differentiable aggregator functions $\text{AGGREGATE}_k, \forall k \in \{1, ..., K\}$; neighborhood function $\mathcal{N} : v \to 2^{\mathcal{V}}$

Output : Vector representations $\mathbf{z}_v$ for all $v \in \mathcal{V}$

$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$; → 用原始特征初始化

for $k = 1...K$ **do**

  for $v \in \mathcal{V}$ **do**

    $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$; → 加总邻居的信息

    $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$

  end

  $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$

end

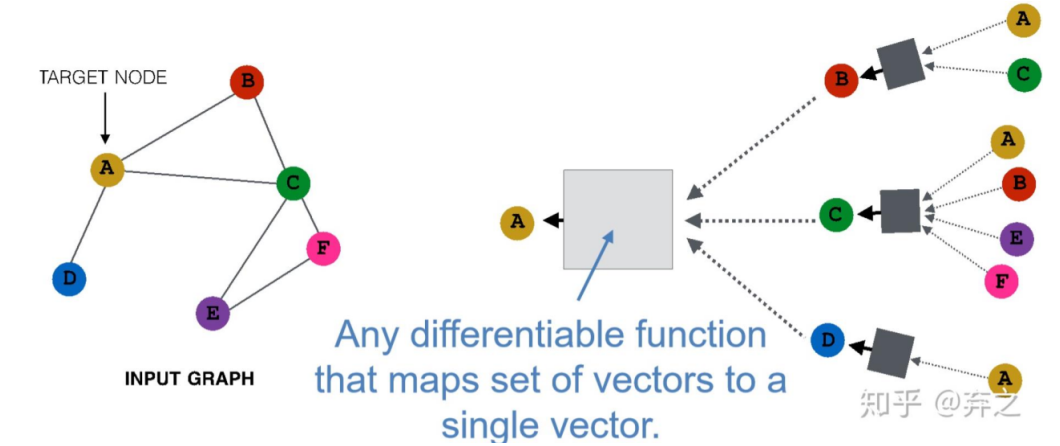
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$

Remark

- 为使得复杂度可控，将邻居数目外生给定为 N，多次有放回随机抽取 N 个
- 反复聚合外生给定的 K 次

前向传播——A toy

K=2


TARGET NODE
A
B
C
D
E
F
INPUT GRAPH
Any differentiable function that maps set of vectors to a single vector.
知乎 @弃之

GraphSAGE——损失函数

$$Loss\ \ Fuction: J_{\mathcal{G}}\left(\mathbf{z}_u\right) = -\log\left(\sigma\left(\mathbf{z}_u^{\top}\mathbf{z}_v\right)\right) - Q \cdot \mathbb{E}_{v_n \sim P_n(v)} \log\left(\sigma\left(-\mathbf{z}_u^{\top}\mathbf{z}_{v_n}\right)\right)$$

- 目标：
  - 相近节点 embedding 表达相似
  - 完全不同节点 embedding 表达差别很大
- Notations: v：在随机游走时，反复出现的点；Q：负样本数目；σ：sigmoid 激活函数
- 特点：训练 embedding 时，同时考虑自身与邻居的特征
- 有监督学习时，采用 Cross-entropy 作为损失函数

GraphSAGE——聚合函数

1. Mean-Aggregator:

$$\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W} \cdot \mathrm{MEAN}\left(\left\{\mathbf{h}_v^{k-1}\right\} \cup \left\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\right\}\right)\right)$$

2. Pool-Aggregator：使用全连接层
3. LSTM-Aggregator：LSTM 本身用于有序向量，而节点是无序的，采用随机排序模拟有序过程

GraphSAGE——模型评价

检验基准：

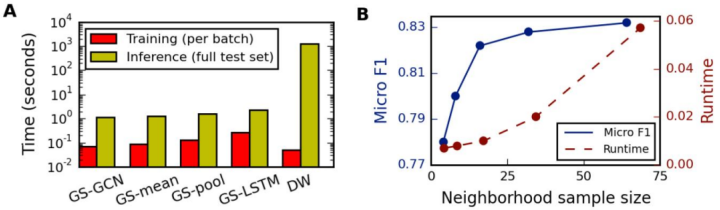
1. WoS 文献库：文献领域分类
2. Reddit：划分帖子所属社区
3. 从蛋白质交互中区别蛋白质功能 (PPI)(Inductive)

	Citation		Reddit		PPI	
Name	Unsup. F1	Sup. F1	Unsup. F1	Sup. F1	Unsup. F1	Sup. F1
Random	0.206	0.206	0.043	0.042	0.396	0.396
Raw features	0.575	0.575	0.585	0.585	0.422	0.422
DeepWalk	0.565	0.565	0.324	0.324	—	—
DeepWalk + features	0.701	0.701	0.691	0.691	—	—
GraphSAGE-GCN	0.742	0.772	**0.908**	0.930	0.465	0.500
GraphSAGE-mean	0.778	0.820	0.897	0.950	0.486	0.598
GraphSAGE-LSTM	0.788	0.832	**0.907**	**0.954**	0.482	**0.612**
GraphSAGE-pool	**0.798**	**0.839**	0.892	0.948	**0.502**	0.600
% gain over feat.	39%	46%	55%	63%	19%	45%

Remark: 使用 F1-Score 作为评价标准

GraphSAGE——模型评价

模型复杂度:



GraphSAGE——模型评价

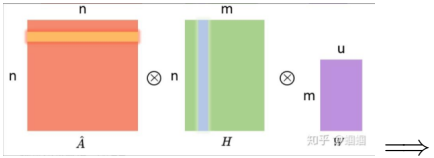
与 GCN 的联系

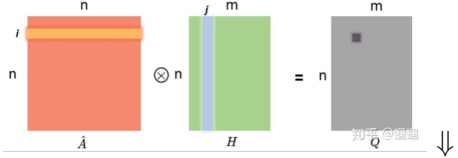
- GCN：$H^{(l+1)} = \sigma\left(\hat{A}H^{(l)}W^{(l)}\right)$
- GraphSage：以 Mean Aggregator 为例

$$\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W} \cdot \mathrm{MEAN}\left(\left\{\mathbf{h}_v^{k-1}\right\} \cup \left\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\right\}\right)\right)$$

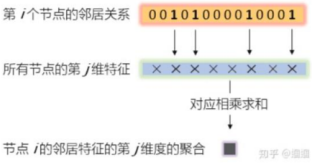
- Remark：都是对邻居特征的的求和，GraphSage 拓展了 GCN 的聚合方式，更加灵活

A toy





1. 相似之处：都是对邻居特征的的求和，邻接矩阵在 GCN 中可视作掩码。
2. GraphSAGE 采用 Mean Aggregator 时，就是 GCN。
3. 由于聚合函数的选取十分灵活，因而作为 GCN 的拓展。



GAT

1. 引入：
   - GCN 无法完成进行 Inductive learning
   - GCN 无法处理有向图
2. 思想：不一定必须在傅里叶域中卷积，直接在原空间域卷积
3. 做法：引入注意力机制

Step 1：Construct Attention Layer

- 注意力系数：计算节点 j 对节点 i 的重要性。
- 首先用共享参数 W 进行特征增强
- 使用 LeakyRelu 作为激活函数（玄学）
- 采用 CONCAT+ 全连接层 + 激活函数 +Softmax 作归一化

$$\alpha_{ij} = \frac{\exp\left(\text{Leaky ReLU}\left(\vec{\mathbf{a}}^T\left[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j\right]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T\left[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k\right]\right)\right)}$$

- Remark：从理论上讲，我们可以将所有点都视为一个点的邻居，不过这样就丢失了图结构特征，同时面临高昂的运算成本，结果可能也不会很好

Step 1: Construct Attention Layer

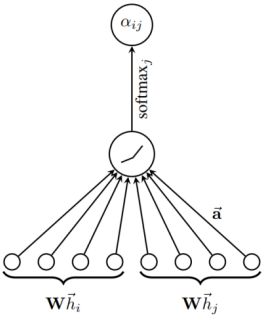


图 1 图解

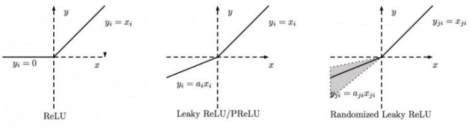


图 2 几种 Relu 形式的比较

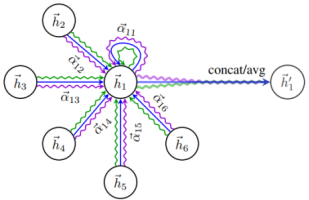
- Remark: 一般而言，LeakyRelu 与 Relu 差别不大，在本文的特定环境下，可能 LeakyRelu 的效果更好

Step 2：Aggregate Neighbors

- 加总所有邻居的重要性

$$h_i'(K) = \|_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}_i} \alpha_{ij}^k W^k h_j\right)$$

- Remark：包含自注意的 multi-head，使用 K 个独立的注意力机制，即 K 个 a 系数矩阵。某点自己的特征对自己的预测能力是越强的，所以自身特征也被注意力机制包含进去。最后，一个节点具有 $K \times F'$ 个特征



GAT 的优势

- 运算高效：
  - 不涉及特征分解与高昂的矩阵运算；
  - 逐节点运算，可以并行，时间复杂度接近 $O(|V|FF' + |E|F')$
  - Q 其中，F 为特征数量，V 和 E 分别为结点数与边数
- 对不同的邻居赋予不同的重要性，GCN 无法办到，GAT 可解释性更好
- 逐节点运算不依赖于全局结构：
  - 可以处理有向图（单向邻居）
  - 可进行 inductive learning

GAT——模型评价

Dataset:

- 区分论文研究领域：Cora, Citeseer, Pubmed (Transductive)
- 根据蛋白质交互图区分蛋白质功能 (PPI) (Inductive)

	Cora	Citeseer	Pubmed	PPI
Task	Transductive	Transductive	Transductive	Inductive
# Nodes	2708 (1 graph)	3327 (1 graph)	19717 (1 graph)	56944 (24 graphs)
# Edges	5429	4732	44338	818716
# Features/Node	1433	3703	500	50
# Classes	7	6	3	121 (multilabel)
# Training Nodes	140	120	60	44906 (20 graphs)
# Validation Nodes	500	500	500	6514 (2 graphs)
# Test Nodes	1000	1000	1000	5524 (2 graphs)

GAT——模型评价

	Transductive		
Method	**Cora**	**Citeseer**	**Pubmed**
MLP	55.1%	46.5%	71.4%
ManiReg (Belkin et al., 2006)	59.5%	60.1%	70.7%
SemiEmb (Weston et al., 2012)	59.0%	59.6%	71.7%
LP (Zhu et al., 2003)	68.0%	45.3%	63.0%
DeepWalk (Perozzi et al., 2014)	67.2%	43.2%	65.3%
ICA (Lu & Getoor, 2003)	75.1%	69.1%	73.9%
Planetoid (Yang et al., 2016)	75.7%	64.7%	77.2%
Chebyshev (Defferrard et al., 2016)	81.2%	69.8%	74.4%
GCN (Kipf & Welling, 2017)	81.5%	70.3%	**79.0%**
MoNet (Monti et al., 2016)	81.7 ± 0.5%	—	78.8 ± 0.3%
GCN-64*	81.4 ± 0.5%	70.9 ± 0.5%	**79.0** ± 0.3%
GAT (ours)	**83.0** ± 0.7%	**72.5** ± 0.7%	**79.0** ± 0.3%

GAT——模型评价

Inductive	
Method	**PPI**
Random	0.396
MLP	0.422
GraphSAGE-GCN (Hamilton et al., 2017)	0.500
GraphSAGE-mean (Hamilton et al., 2017)	0.598
GraphSAGE-LSTM (Hamilton et al., 2017)	0.612
GraphSAGE-pool (Hamilton et al., 2017)	0.600
GraphSAGE*	0.768
Const-GAT (ours)	0.934 ± 0.006
GAT (ours)	$\mathbf{0.973} \pm 0.002$

DropEdge

- 引入：
  - 图卷积网络很难加深
  - GCN 无法处理有向图
- 结论：过拟合和过平滑制约了图神经网络的加深
  - Remark: 过平滑是图神经网络中特有的问题，由于 GCN 的思想是聚合邻居节点和节点自身的信息来学习节点的表示，所以随着网络层数的加深，节点的表示有趋同性，节点表示的可区分性会变差。
  - 随着层数的不断加深，所有节点的表示最终会收敛到一个固定点，得到的节点表示和输入特征无关，而且还会导致梯度消失。
- 应用：可用于优化 GCN、ResGCN、JKNet、GraphSAGE 等图卷积网络模型

DropEdge——Methodology

以 GCN 为例:

- Step 1: 原始传播方式：$\boldsymbol{H}^{(l+1)} = \sigma\left(\hat{\boldsymbol{A}}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)}\right)$

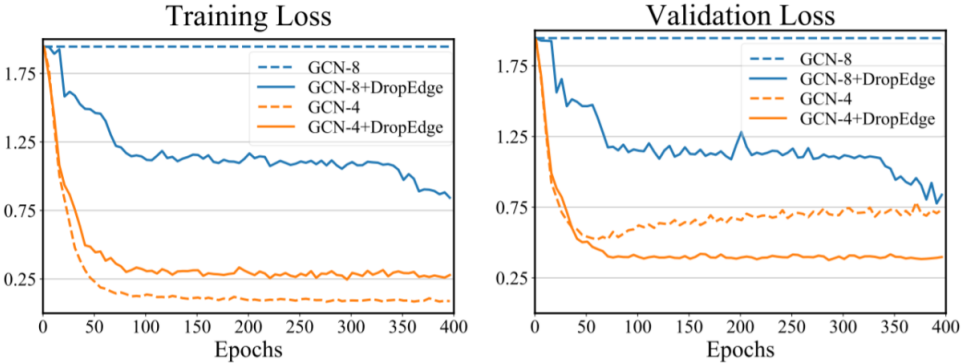
  Remark: $\hat{A} = \hat{D}^{-1/2}(A + I)\hat{D}^{-1/2}$;$D$ 为度矩阵，用来归一化，A 为邻接矩阵
- 每次训练时 (At each epoch)，随机删除原始图中固定比例 (p) 的边，得到 A_{drop} 再用 Step 1 的方法归一化得到 $\hat{A}_{\text{drop}}$ 以 $\hat{A}_{\text{drop}}$ 替换 $\hat{A}$ 进行传播

$$A_{\text{drop}} = A - A'$$

DropEdge——Implications

- DropEdge 机制对输入的邻接矩阵进行扰动, 即随机变形，可以视作是数据增强的方式，类似对图像旋转、剪切，翻转。
- DropEdge 是无偏的数据增强方式：在以概率 p 删边后，又重新进行了归一化处理，因而是无偏数据增强
- DropEdge 只在每次训练开始时随机丢弃边，还可以逐层调用 DropEdge 机制，即在每一层训练时都进行 DropEdge，这样可以带来更大的随机性，减少过拟合

DropEdge——Preliminary Results



DropEdge——Preventing Over-smoothing

Definition **1** (subspace). Let $\mathcal{M}: = \left\{\boldsymbol{EC} \mid \boldsymbol{C} \in \mathbb{R}^{M \times C}\right\}$ be an M -dimensional subspace in $\mathbb{R}^{N \times C}$, where $\boldsymbol{E} \in \mathbb{R}^{N \times M}$ is orthogonal, i.e. $\boldsymbol{E}^{\mathrm{T}} \boldsymbol{E} = \boldsymbol{I}_M$, and $\mathrm{M} \leq \mathrm{N}$.

DropEdge——Preventing Over-smoothing

DropEdge 能缓解过平滑问题:

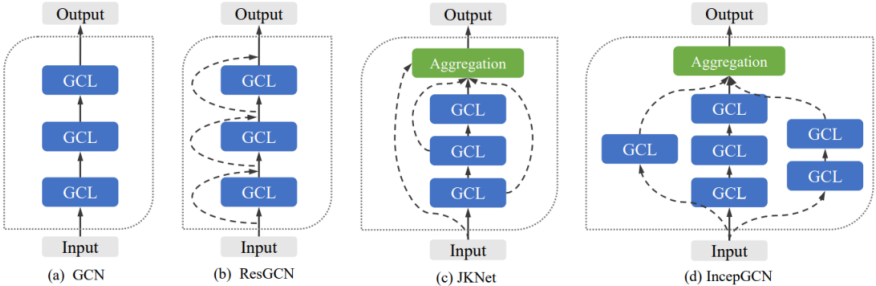
- 减少节点间的联系，减缓收敛速度
- 减少原始空间与收敛子空间的信息损失，例如 N–dim(M)，当 GCN 足够深时，大量信息被丢弃

Comparison

- DropEdge VS. Dropout (Hinton et al., 2012)：Dropout 随机丢弃特征，可以缓解过拟合但无法缓解过平滑问题。DropEdge 可以看作 Dropout 在图模型中的推广
- DropEdge VS. DropNode(Hamilton et al., 2017; Chen et al.,2018; Huang et al., 2018)：DropNode 面向节点且无向，把与该节点相连的边一并删掉。DropEdge 面向边，更加灵活且高效，层数加深时不会增加 layer size(GraphSAGE)，也不需要递归采样 (AS-GCN)，所有边可以并行采样
- DropEdge VS. Graph-Sparsification（Eppstein et al., 1997): 图稀疏性的目标是去除掉不必要的边，对图进行压缩，同时尽可能地保留原始输入图中的信息，需要优化，繁琐耗时。而 DropEdge 没有目标函数，每次训练时随机丢弃

DropEdge——模型评价

- Dataset:
  - 区分论文研究领域：Cora, Citeseer, Pubmed (transductive)
  - 预测帖子属于哪个社区：Reddit (inductive)
- Baseline Models：GCN、ResGCN、JKNet、IncepGCN、GraphSAGE



DropEdge——模型评价

Table 1: Testing accuracy (%) comparisons on different backbones w and w/o DropEdge.

Dataset	Backbone	2 layers		8 layers		32 layers	
		Orignal	DropEdge	Orignal	DropEdge	Orignal	DropEdge
Cora	GCN	86.10	**86.50**	78.70	**85.80**	71.60	**74.60**
	ResGCN	-	-	85.40	**86.90**	85.10	**86.80**
	JKNet	-	-	86.70	**87.80**	87.10	**87.60**
	IncepGCN	-	-	86.70	**88.20**	87.40	**87.70**
	GraphSAGE	87.80	**88.10**	84.30	**87.10**	31.90	**32.20**
Citeseer	GCN	75.90	**78.70**	74.60	**77.20**	59.20	**61.40**
	ResGCN	-	-	77.80	**78.80**	74.40	**77.90**
	JKNet	-	-	79.20	**80.20**	71.70	**80.00**
	IncepGCN	-	-	79.60	**80.50**	72.60	**80.30**
	GraphSAGE	78.40	**80.00**	74.10	**77.10**	37.00	**53.60**
Pubmed	GCN	90.20	**91.20**	90.10	**90.90**	84.60	**86.20**
	ResGCN	-	-	89.60	**90.50**	90.20	**91.10**
	JKNet	-	-	90.60	**91.20**	89.20	**91.30**
	IncepGCN	-	-	90.20	**91.50**	OOM	**90.50**
	GraphSAGE	90.10	**90.70**	90.20	**91.70**	41.30	**47.90**
Reddit	GCN	96.11	**96.13**	96.17	**96.48**	45.55	**50.51**
	ResGCN	-	-	96.37	**96.46**	93.93	**94.27**
	JKNet	-	-	96.82	**97.02**	OOM	OOM
	IncepGCN	-	-	96.43	**96.87**	OOM	OOM
	GraphSAGE	96.22	**96.28**	96.38	**96.42**	96.43	**96.47**

DropEdge——模型评价

对比前沿模型:

Table 2: Accuracy (%) comparisons with SOTAs. The number in parenthesis denotes the network depth for the models with DropEdge.

	Transductive			Inductive
	Cora	Citeseer	Pubmed	Reddit
GCN	86.64	79.34	90.22	95.68
FastGCN	85.00	77.60	88.00	93.70
ASGCN	87.44	79.66	90.60	96.27
GraphSAGE	82.20	71.40	87.10	94.32
GCN+DropEdge	87.60(4)	79.20(4)	91.30(4)	96.71(4)
ResGCN+DropEdge	87.00(4)	79.40(16)	91.10(32)	96.48(16)
JKNet+DropEdge	88.00(16)	80.20(8)	91.60(64)	**97.02(8)**
IncepGCN+DropEdge	**88.20(8)**	**80.50(8)**	91.60(4)	96.87(8)
GraphSAGE+DropEdge	88.10(4)	80.00(2)	**91.70(8)**	96.54(4)

DropEdge——模型评价

过平滑程度:

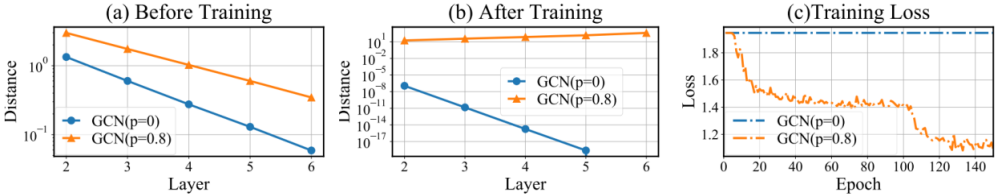
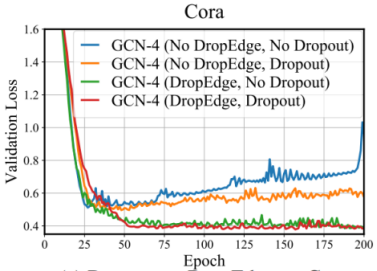


Figure 3: Analysis on over-smoothing. Smaller distance means more serious over-smoothing.

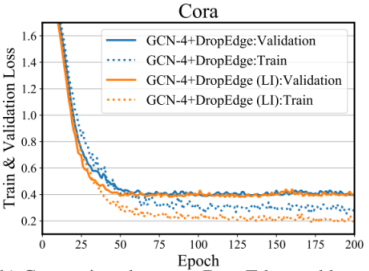
Remark：计算当前层与上一层输出结果的欧式距离，距离越小代表过平滑问题越严重

DropEdge——模型评价

- 与 Dropout 的兼容性
- 一次性 DropEdge 与逐层 (Layer-Wise) DropEdge 的差异



(a) Dropout vs DropEdge on Cora.



(b) Comparison between DropEdge and layer-wise (LW) DropEdge.

DropEdge——Conclusion

- DropEdge 机制有助于加深图卷积网络，用于节点分类任务
- DropEdge 缓解了 GCN 中过拟合与过平滑问题
- 在现有模型下应用 DropEdge 机制能提升原模型性能

参考文献 I

Karsten Borgwardt. and Elisabetta Ghisu.
Graph Kernels, 2020.
https://arxiv.org/pdf/2011.03854v2.pdf

- 刘忠雨. and 李彦霖. and 周洋
  深入浅出图神经网络.
  机械工业出版社,2019

Petar and Cucurull, Guillem and Casanova, Arantxa and Romero, Adriana and Lio, Pietro and Bengio, Yoshua
Graph attention networks, 2017.
https://arxiv.org/pdf/1710.10903.pdf

参考文献 II


Kipf, Thomas N and Welling, Max
Semi-supervised classification with graph convolutional networks, 2016.
https://arxiv.org/pdf/1609.02907.pdf?source=post_page----------------------------

[5] 孙建东
《*Semi-Supervised Classification with Graph Convolutional Networks*》阅读笔记, 2018, 知乎.
https://zhuanlan.zhihu.com/p/31067515

Thanks!