



南開大學
Nankai University

Graph neural networks for text classification

Z. Shi

Nankai University

2021 年 1 月 7 日



目录

- 1 图神经网络
- 2 图卷积网络：谱方法
- 3 GraphSAGE and GAT
- 4 Experiments and Further works



目录

1 图神经网络

2 图卷积网络：谱方法

3 GraphSAGE and GAT

4 Experiments and Further works



Network Embedding

- Network Analysis
 - PageRank(Brin and Page, 1998)
- Matrix Factorization
 - Factorization of Lapacian of adjacency matrix(Belkin and Niyogi, 2002)
- Random walk
 - Deepwalk(Perozzi et al., 2014) considers the random walk paths as sentences and implements SkipGram to learn embedding of each node.
 - Node2vec(Grover and Leskovec, 2016) designs an adaptive random walk to sample the first and second neighborhood nodes.



Deep Learning for Network Embedding

- LINE (large-scale information network embedding)(Tang et al., 2015b)
 - preserves the first and second order proximities, and minimize the KL divergence between them.
- SDNE (structural deep network embedding)(Wang et al., 2016)
 - uses the deep autoencoder with multiple nonlinear layers to preserve the neighbor structures of nodes,
 - adopts Laplacian Eigenmap to exploit the first-order and second-order proximities jointly into the learning process as penalty.
- Graph Neural Network and its variants
 - CNN 的平移不变性在非欧结构数据上不适用
 - 传统的网络表示学习方法计算量大、泛化能力弱



图神经网络 (graph neural network, GNN) I

- 目标：对图的顶点数据、边数据和子图数据进行降维，在拓扑图上提取空间特征来进行学习
- 综述 (Zhang et al., 2018; Zhou et al., 2018)
- 学习一个包含每个顶点 v 的邻接信息的嵌入状态 \mathbf{h}_v

$$\mathbf{h}_v = f(\mathbf{x}_v, \mathbf{x}_{co[v]}, \mathbf{h}_{ne[v]}, \mathbf{x}_{ne[v]}) \quad (1)$$

其中 $\mathbf{x}_v, \mathbf{x}_{co[v]}$ 表示 v 及其边的特征， $\mathbf{h}_{ne[v]}, \mathbf{x}_{ne[v]}$ 表示 v 的邻接的状态和特征

- 输出

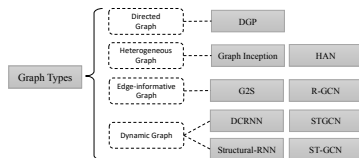
$$\mathbf{z}_v = g(\mathbf{h}_v, \mathbf{x}_v) \quad (2)$$

- 假设目标信息为 t_v ，则损失函数

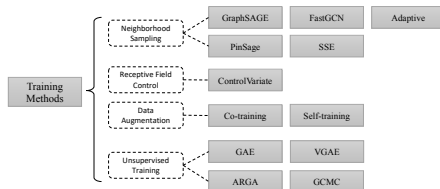
$$MSE = \sum_i^p ||t_i - z_i||^2 \text{ or } CE = - \sum_i^p t_i \log z_i \quad (3)$$



图神经网络 (graph neural network, GNN) II



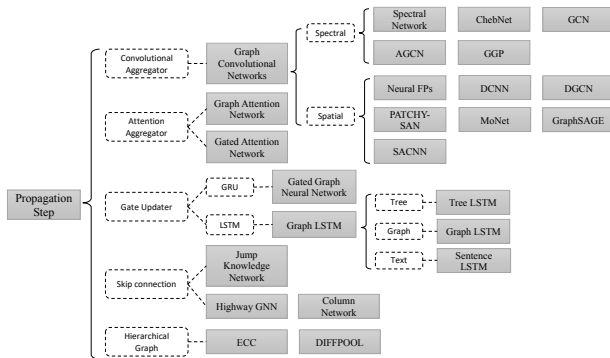
(a) Graph Types



(b) Training Methods



图神经网络 (graph neural network, GNN) III



(c) Propagation Steps

图 1: An overview of variants of graph neural networks.



目录

- 1 图神经网络
- 2 图卷积网络：谱方法
- 3 GraphSAGE and GAT
- 4 Experiments and Further works

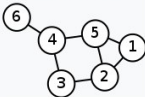


Laplacian Matrix I

Definition 1 (Laplacian Matrix)

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \quad (4)$$

其中 \mathbf{D} 表示度矩阵 (degree matrix), 即以每个顶点的度为对角元的对角矩阵, 所谓度 (degree) 即由该点发出的边之数量。 \mathbf{A} 表示邻接矩阵 (adjacency matrix)。

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

拉普拉斯算子

$$\nabla^2 f = \nabla \cdot \nabla f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$



(5)

Laplacian Matrix II

- 离散化

$$\nabla^2 f = f(x+1, y) + f(x, y+1) + f(x-1, y) + f(x, y-1) - 4f(x, y) \quad (6)$$

写成卷积核形式

$$L = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad (7)$$

- 把 $f(x+1, y), f(x, y+1), f(x-1, y), f(x, y-1), f(x, y)$ 看做一个图的节点 v_1, v_2, v_3, v_4, v_5 的映射，那么 v_5 分别与 v_1, v_2, v_3, v_4 相邻，这样，拉普拉斯算子就可以用邻接矩阵和度矩阵的差来表达。
- 在图论中，我们习惯用度矩阵减去邻接矩阵（这保证了拉普拉斯矩阵是半正定的），即 $L = D - A$



Laplacian Matrix III

- 正则化的拉普拉斯矩阵 (Symmetric normalized Laplacian)

$$\mathbf{L} \leftarrow \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2} \quad (8)$$

即

$$\mathbf{L} = \mathbf{I}_N - \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}} = \mathbf{U} \mathbf{\Lambda} \mathbf{U}^T \quad (9)$$

所以

$$\mathbf{L} = \begin{cases} 1 & i = j \text{ and } \deg(v_i) \neq 0 \\ -\frac{1}{\sqrt{\deg(v_i) \deg(v_j)}} & i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise} \end{cases} \quad (10)$$



Fourier Transform on Graph I

- 傅里叶变换将时域信号变换到频域，在时域信号中的卷积运算，变为频域信号中的相乘。

$$F(\omega) = \mathcal{F}[f(t)] = \int f(t)e^{-j\omega t} dt \quad (11)$$

- 空间域 (spatial domain) 的图卷积运算变换为频谱域 (spectral domain) 的矩阵相乘。
- $e^{-j\omega t}$ 既是拉普拉斯算子的特征函数，又是傅里叶变换的基

$$\nabla^2 e^{-j\omega t} = \frac{\partial^2}{\partial t^2} e^{-j\omega t} = -\omega^2 e^{-j\omega t} \quad (12)$$



Fourier Transform on Graph II

拉普拉斯矩阵与图的傅里叶变换

- 对拉普拉斯矩阵进行特征值分解（谱分解）
- 图上的拉普拉斯矩阵，相当于离散拉普拉斯算子。
- 正则化拉普拉斯矩阵的特征列向量 u_l 是一组单位正交基，相当于拉普拉斯算子的特征函数 $e^{-j\omega t}$ ，也相当于对图进行傅里叶变换的一组基。
- 正则化拉普拉斯矩阵的特征值均为非负的，相当于图的“频率”。



Fourier Transform on Graph III

FT and IFT on Graph

$$\mathcal{F}(\lambda_l) = \hat{f}(\lambda_l) = \sum_{i=1}^N f(i)u_l(i) \quad (13)$$

$$\Rightarrow \begin{pmatrix} \hat{f}(\lambda_1) \\ \hat{f}(\lambda_2) \\ \vdots \\ \hat{f}(\lambda_N) \end{pmatrix} = \begin{pmatrix} u_1(1) & u_1(2) & \dots & u_1(N) \\ u_2(1) & u_2(2) & \dots & u_2(N) \\ \vdots & \vdots & \ddots & \vdots \\ u_N(1) & u_N(2) & \dots & u_N(N) \end{pmatrix} \begin{pmatrix} f(1) \\ f(2) \\ \vdots \\ f(N) \end{pmatrix} \quad (14)$$

Therefore

$$\hat{f} = U^T f \text{ and } f = U \hat{f} \quad (15)$$



图卷积网络 (graph convolutional network, GCN) I

- 卷积核 (一个滤波器) 的傅里叶变换是一个对角矩阵 $g_\theta = \text{diag}(\theta_i(\lambda_i))$, 与信号 \mathbf{x} 做卷积, \mathbf{x} 经过傅里叶变换 $\hat{\mathbf{x}} = U^T \mathbf{x}$, 与 g_θ 相乘后再做傅里叶逆变换, 即 $U g_\theta U^T \mathbf{x}$

$$\mathbf{g}_\theta \star \mathbf{x} = U \mathbf{g}_\theta(\Lambda) U^T \mathbf{x} \quad (16)$$

- $\mathbf{g}_\theta(\Lambda)$ 可以通过展开为切比雪夫多项式 $T_k(x)$ 来估计到 k 阶 (Hammond et al., 2011)

$$\mathbf{g}_\theta \star \mathbf{x} \approx \sum_{k=0}^K \theta_k \mathbf{T}_k(\tilde{\mathbf{L}}) \mathbf{x} \quad (17)$$

$$\tilde{\mathbf{L}} = \frac{2}{\lambda_{\max}} \mathbf{L} - \mathbf{I}_N \quad (18)$$

其中 λ_{\max} 是 \mathbf{L} 的最大特征值。



图卷积网络 (graph convolutional network, GCN) II

Definition 2 (切比雪夫多项式)

$$\mathbf{T}_k(\mathbf{x}) = 2\mathbf{x}\mathbf{T}_{k-1}(\mathbf{x}) - \mathbf{T}_{k-2}(\mathbf{x}), \mathbf{T}_0(\mathbf{x}) = 1, \mathbf{T}_1(\mathbf{x}) = \mathbf{x} \quad (19)$$

- 一般可以认为 $\lambda_{max} \approx 2$ (Chung, 2012), 且将切比雪夫多项式仅展开到一阶 ($k=1$)

$$\mathbf{g}_{\theta'} \star \mathbf{x} \approx \theta'_0 \mathbf{x} + \theta'_1 (\mathbf{L} - \mathbf{I}_N) \mathbf{x} = \theta'_0 \mathbf{x} - \theta'_1 \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}} \mathbf{x} \quad (20)$$

不妨令 $\theta = \theta'_0 = -\theta'_1$ 则

$$\mathbf{g}_{\theta} \star \mathbf{x} \approx \theta \left(\mathbf{I}_N + \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}} \right) \mathbf{x} \quad (21)$$

- 使用重正则化技巧, 也就是给图加上自环, 这样新的邻接矩阵 $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$, 度矩阵 $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$, 且 $\mathbf{I}_N + \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}} \rightarrow \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}}$



图卷积网络 (graph convolutional network, GCN) III

Graph Convolutional Network

(Kipf and Welling, 2017) 提出了目前通用的 GCN

$$\mathbf{Z} = \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{X} \Theta \quad (22)$$

Add nonlinear activation

$$\mathbf{H}^{(l+1)} = \sigma \left(\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^{(l)} \mathbf{W}^{(l)} \right) \quad (23)$$

where $\mathbf{H}^0 = \mathbf{X}$



图卷积网络 (graph convolutional network, GCN) IV

Citation Network(Citeseer, Cora and Pubmed)

- Cora 数据集 (Prithviraj et al., 2008) 共 2708 个样本点，每个样本点都是一篇论文，所有样本点被分为 7 个类别：1) 基于案例；2) 遗传算法；3) 神经网络；4) 概率方法；5) 强化学习；6) 规则学习；7) 理论
- 每篇论文都由一个 1433 维的词向量表示其特征向量，词向量的每个元素都对应一个词，且该元素只有 0 或 1 两个取值。取 0 表示该元素对应的词不在论文中，取 1 表示在论文中。
- 论文的引用关系是图的边，图上没有孤立点。
- 只有少量样本点 (Cora 中为 140 个点) 被打上了类别标签，称为半监督分类 (semi-supervised classification)



图卷积网络 (graph convolutional network, GCN) V

A two-layer GCN

Pre-processing $\hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$

$$Z = f(X, A) = \text{softmax}\left(\hat{A} \text{ReLU}\left(\hat{A} X W^{(0)}\right) W^{(1)}\right). \quad (24)$$

$$\mathcal{L} = - \sum_{l \in \mathcal{Y}_L} \sum_{f=1}^F Y_{lf} \ln Z_{lf} \quad (25)$$



Performance

Table 2: Summary of results in terms of classification accuracy (in percent).

Method	Citeseer	Cora	Pubmed	NELL
ManiReg [3]	60.1	59.5	70.7	21.8
SemiEmb [28]	59.6	59.0	71.1	26.7
LP [32]	45.3	68.0	63.0	26.5
DeepWalk [22]	43.2	67.2	65.3	58.1
ICA [18]	69.1	75.1	73.9	23.1
Planetoid* [29]	64.7 (26s)	75.7 (13s)	77.2 (25s)	61.9 (185s)
GCN (this paper)	70.3 (7s)	81.5 (4s)	79.0 (38s)	66.0 (48s)
GCN (rand. splits)	67.9 ± 0.5	80.1 ± 0.5	78.9 ± 0.7	58.4 ± 1.7

Table 3: Comparison of propagation models.

Description	Propagation model	Citeseer	Cora	Pubmed
Chebyshev filter (Eq. 5)	$K = 3$	69.8	79.5	74.4
	$K = 2$	69.6	81.2	73.8
1 st -order model (Eq. 6)	$X\Theta_0 + D^{-\frac{1}{2}}AD^{-\frac{1}{2}}X\Theta_1$	68.3	80.0	77.5
Single parameter (Eq. 7)	$(I_N + D^{-\frac{1}{2}}AD^{-\frac{1}{2}})X\Theta$	69.3	79.2	77.4
Renormalization trick (Eq. 8)	$\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X\Theta$	70.3	81.5	79.0
1 st -order term only	$D^{-\frac{1}{2}}AD^{-\frac{1}{2}}X\Theta$	68.7	80.5	77.8
Multi-layer perceptron	$X\Theta$	46.5	55.1	71.4



Relation to Weisfeiler-Lehman Algorithm I

1-dim Weisfeiler-Lehman (WL-1) algorithm

$$h_i^{(t+1)} \leftarrow \text{hash} \left(\sum_{j \in \mathcal{N}_i} h_j^{(t)} \right) \quad (26)$$

Here, $h_i^{(t)}$ denotes the coloring (label assignment) of node v_i (at iteration t) and \mathcal{N}_i is its set of neighboring node indices.
(Weisfeiler and Leman, 1968)

- Replace the hash function with a neural network layer-like differentiable function with trainable parameters as follows:

$$h_i^{(l+1)} = \sigma \left(\sum_{j \in \mathcal{N}_i} \frac{1}{c_{ij}} h_j^{(l)} W^{(l)} \right), \quad (27)$$



Relation to Weisfeiler-Lehman Algorithm II

- By choosing $c_{ij} = \sqrt{\tilde{d}_i \tilde{d}_j}$, we recover the propagation rule of GCN.
- Even an untrained GCN model with random weights can serve as a powerful feature extractor for nodes in a graph.

Zachary's karate club

- (Zachary, 1976) 34 nodes (people), 154 edges (links), 4 classes, labeled 1 node per class.
- Featureless approach by setting $X = I$ (one-hot)
- Random weight matrices comparable to Deepwalk



Model depth of GCN I

- Residual

$$H^{(l+1)} = \sigma \left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)} \right) + H^{(l)}. \quad (28)$$

- Best results are obtained with a 2- or 3-layer model.
- For models deeper than 7 layers, training without the use of residual connections(He et al., 2016) can become difficult, as the effective context size for each node increases by the size of its K^{th} -order neighborhood (for a model with K layers) with each additional layer.
- Overfitting can become an issue as the number of parameters increases with model depth.



Model depth of GCN II

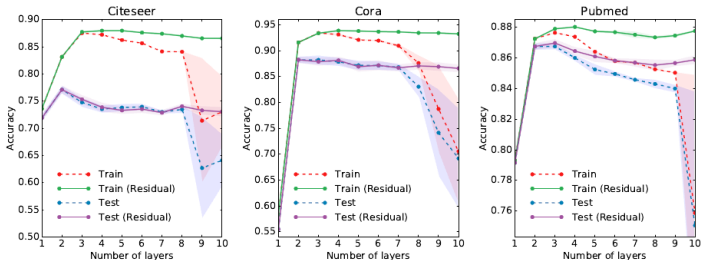


Figure 5: Influence of model depth (number of layers) on classification performance. Markers denote mean classification accuracy (training vs. testing) for 5-fold cross-validation. Shaded areas denote standard error. We show results both for a standard GCN model (dashed lines) and a model with added residual connections (He et al., 2016) between hidden layers (solid lines).



目录

- 1 图神经网络
- 2 图卷积网络：谱方法
- 3 GraphSAGE and GAT
- 4 Experiments and Further works



GraphSAGE I

- Spectral GCN is transductive learning (直推式学习, embedding 唯一确定), GraphSAGE is inductive learning (归纳式学习, embedding 随着新加入的 link 而改变) .
- 谱方法 (spectrum method) 依赖于对拉普拉斯矩阵进行特征值分析, 尽管能捕获全局信息, 却削弱了泛化能力, 需要把所有节点都参与训练来得到 embedding.
- 空间域方法 (spatial method) 直接在图上定义卷积, 其主要挑战是既要对不同大小的邻域定义卷积, 又要保证 CNN 的局部不变性, 最常用的方法是 GraphSAGE(graph sampling and aggregating)(Hamilton et al., 2017), 即对顶点的局部邻域进行采样和聚合来生成 embedding.



GraphSAGE II

Aggregator

均值融合 直接取邻居节点的平均。

LSTM 融合 交换节点邻域，引入 LSTM(Hochreiter and Schmidhuber, 1997) 操作这一无序集合。

池化融合 隐藏状态通过全连接层后，进行 max-pooling 操作。

$$\mathbf{h}_{\mathcal{N}_v}^t = \max(\{\sigma(\mathbf{W}_{\text{pool}}\mathbf{h}_u^{t-1} + \mathbf{b}), \forall u \in \mathcal{N}_v\}) \quad (29)$$



GraphSAGE III

Message passing neural network

Message passing neural network is a general framework for supervised learning on graphs

Aggregator $\mathbf{m}_v^{t+1} = \sum_{w \in \mathcal{N}_v} M_t(\mathbf{h}_v^t, \mathbf{h}_w^t, \mathbf{e}_{vw})$

Updater $\mathbf{h}_v^{t+1} = U_t(\mathbf{h}_v^t, \mathbf{m}_v^{t+1})$

Readout $\hat{\mathbf{y}} = R(\{\mathbf{h}_v^T | v \in G\})$



GraphSAGE IV

Algorithm 1 GraphSAGE embedding generation

- 1: $h_0^v = x_v$
 - 2: **for** $k \in [1, K]$ **do**
 - 3: **for** $v \in \mathcal{V}$ **do**
 - 4: Aggregator: $\mathbf{h}_{\mathcal{N}_v}^t = \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}_v\})$
 - 5: Updater: $\mathbf{h}_v^k = \sigma(\mathbf{W}^k \cdot [\mathbf{h}_v^{k-1} \parallel \mathbf{h}_{\mathcal{N}_v}^k])$
 - 6: **end for**
 - 7: Readout1: $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2$
 - 8: **end for**
 - 9: Readout2: $\mathbf{z}_v \leftarrow \mathbf{h}_v^K$
-



GraphSAGE V

GraphSAGE-GCN

- 将算法1的第 4, 5 步改为

$$\mathbf{h}_v^t = \sigma(\mathbf{W} \cdot \text{mean}(\{\mathbf{h}_v^{t-1}\} \cup \{\mathbf{h}_u^{t-1}, \forall u \in \mathcal{N}_v\})) \quad (30)$$

- 相比于其他三种聚合器, (30)式相当于一种残差连接。
- 相比于基本的 GCN

$$\bar{\mathbf{h}}_i^{(k)} \leftarrow \frac{1}{d_i + 1} \mathbf{h}_i^{(k-1)} + \sum_{j=1}^n \frac{1}{\sqrt{(d_i + 1)(d_j + 1)}} \mathbf{h}_j^{(k-1)}, \forall v_j \in N_v \quad (31)$$

(30)式相当于 inductive GCN.

$$\bar{\mathbf{h}}_i^{(k)} \leftarrow \frac{1}{d_i + 1} \left(\mathbf{h}_i^{(k-1)} + \sum_{j=1}^n \mathbf{h}_j^{(k-1)} \right), \forall v_j \in N_v \quad (32)$$



Performance

Table 1: Prediction results for the three datasets (micro-averaged F1 scores). Results for unsupervised and fully supervised GraphSAGE are shown. Analogous trends hold for macro-averaged scores.

Name	Citation		Reddit		PPI	
	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%

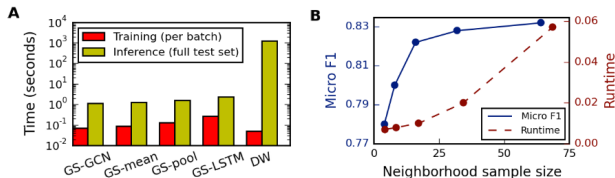


Figure 2: **A:** Timing experiments on Reddit data, with training batches of size 512 and inference on the full test set (79,534 nodes). **B:** Model performance with respect to the size of the sampled neighborhood, where the “neighborhood sample size” refers to the number of neighbors sampled at each depth for $K = 2$ with $S_1 = S_2$ (on the citation data using GraphSAGE-mean).



Graph Attention Network, GAT I

- (Velickovic et al., 2018) presents graph attention networks (GAT), leveraging masked self-attentional layers on graph convolution.

$$z_i^{(l)} = W^{(l)} h_i^{(l)} \quad (33)$$

$$e_{ij}^{(l)} = \text{LeakyReLU}(\vec{a}^{(l)T} (z_i^{(l)} || z_j^{(l)})) \quad (34)$$

$$\alpha_{ij}^{(l)} = \frac{\exp(e_{ij}^{(l)})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik}^{(l)})} \quad (35)$$

$$h_i^{(l+1)} = \sigma \left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} z_j^{(l)} \right) \quad (36)$$



Graph Attention Network, GAT II

- Equation (33) is a linear transformation of the lower layer embedding $h_i^{(l)}$.
- Equation (34) computes a pair-wise unnormalized attention score between two neighbors. Here, it first concatenates the z embeddings of the two nodes, then takes a dot product of it and a learnable weight vector $a^{(l)}$, and applies a LeakyReLU in the end.
- Equation (35) applies a softmax to normalize the attention scores on each node's in-coming edges.
- Equation (36) is similar to GCN. The embeddings from neighbors are aggregated together, scaled by the attention scores.



目录

- 1 图神经网络
- 2 图卷积网络：谱方法
- 3 GraphSAGE and GAT
- 4 Experiments and Further works



Libraries about GCN

- PyTorch(Paszke et al., 2019)
- DGL(Deep Graph Library)(Wang et al., 2019a)
- PyG(PyTorch Geometric)(Paszke et al., 2019)



Methodology of TextGCN I

Definition 3 (PMI)

The PMI value of a word pair i, j is computed as

$$\text{PMI}(i, j) = \log \frac{p(i, j)}{p(i)p(j)} \quad (37)$$

$$p(i, j) = \frac{\#W(i, j)}{\#W} \quad (38)$$

$$p(i) = \frac{\#W(i)}{\#W} \quad (39)$$

We only add edges between word pairs with positive PMI values.



Methodology of TextGCN II

Definition 4 (TF-IDF)

- 词频 (term frequency, tf): 对于词语 t_i

$$\text{tf}_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad (40)$$

其中 $n_{i,j}$ 是该词在文章 d_j 中的出现次数, 而分母则是文章的总词数。

- 逆向文件频率 (inverse document frequency, idf):

$$\text{idf}_i = \lg \frac{|D|}{1 + |\{j : t_i \in d_j\}|} \quad (41)$$

其中 $|D|$ 表示语料库中的文件总数, $|\{j : t_i \in d_j\}|$ 表示包含词语 t_i 的文件数目

$$\text{tfidf}_{i,j} = \text{tf}_{i,j} \times \text{idf}_i \quad (42)$$



Methodology of TextGCN III

A heterogeneous text network

$$A_{ij} = \begin{cases} \text{PMI}(i, j) & i, j \text{ are words, } \text{PMI}(i, j) > 0 \\ \text{TF-IDF}_{ij} & i \text{ is document, } j \text{ is word} \\ 1 & i = j \\ 0 & \text{otherwise} \end{cases} \quad (43)$$



测试结果

- 使用 MR 数据集 (Tang et al., 2015a) 进行文本分类 (节点分类)。

表 1: 测试结果

算法	准确率 (mr)	Macro-F1 (mr)
GCN (原文献)	0.7552	0.7548
GCN	0.7541	0.7536
SAGEMean	0.7642	0.7637
GAT	0.7665	0.7664



Further works and applications I

- Graph Transformer
 - Multi-head dot-product attention(MHDP) in Transformer(Vaswani et al., 2017) can be interpreted as implementing updates and aggregations over a graph.
 - seq2seq, encoder-decoder
 - From node j to i , $q = W_q h_i, k = W_k h_j, v = W_v h_j$ and attention weights of edges

$$A = \text{softmax} \left(\frac{QK^T}{\sqrt{d_i}} \right) V \quad (44)$$

- Unsupervised learning
 - Unsupervised loss in GraphSAGE to learn a representation for each node.
 - Variation graph auto-encoder(Kipf and Welling, 2016) for representation learning and link prediction (graph reconstruction)



Further works and applications II

- Graph classification (graph pooling)
 - The paper (Xu et al., 2019b) analyzes how powerful are GNNs, and learns a representation for a graph
- Theoretical works and graph signal processing
 - Simplify graph convolution(Wu et al., 2019)
 - Graph wavelet neural networks(Xu et al., 2019a)
- Applications
 - MHDPA with graph neural networks on reinforcement learning tasks like playing StarCraft II(Zambaldi et al., 2018) and robot manipulation(Wilson and Hermans, 2019).
 - Text classification on Heterogeneous graph, and heterograph attention on social network(Wang et al., 2019b; Yao et al., 2019)
 - Computer Vision: spatial-temporal GCN(Yan et al., 2018) with OpenPose(Cao et al., 2017)
 - Bioinformatics (Zitnik et al., 2018)



Reference I

- Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in Neural Information Processing Systems*, 14(6):585–591, 2002.
- Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.
- Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017.
- Fan Chung. *Spectral Graph Theory*. AMS, 2012.
- Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *the 22nd ACM SIGKDD International Conference*, 2016.



Reference II

- William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.
- D. Hammond, P. Vandergheynst, and R. Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Sun Jian. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.
- Thomas N. Kipf and Max Welling. Variational graph auto-encoders. In *NeurIPS*, 2016.



Reference III

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *the 20th ACM SIGKDD International Conference*, 2014.

Sen Prithviraj, Namata Galileo, Bilgic Mustafa, Getoor Lise, Galligher Brian, and Eliassi-Rad Tina. Collective classification in network data. *Ai Magazine*, 29(3):93, 2008.



Reference IV

- Jian Tang, Meng Qu, and Qiaozhu Mei. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *KDD 2015*, 2015a.
- Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *24th International Conference on World Wide Web, WWW 2015*, 2015b.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.
- Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.



Reference V

Daixin Wang, Cui Peng, and Wenwu Zhu. Structural deep network embedding. In *the 22nd ACM SIGKDD International Conference*, 2016.

Minjie Wang, Lingfan Yu, Da Zheng, Quan Gan, Yu Gai, Zihao Ye, Mufei Li, Jinjing Zhou, Qi Huang, Chao Ma, Ziyue Huang, Qipeng Guo, Hao Zhang, Haibin Lin, Junbo Zhao, Jinyang Li, Alexander J Smola, and Zheng Zhang. Deep graph library: Towards efficient and scalable deep learning on graphs. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019a.

Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Peng Cui, P. Yu, and Yanfang Ye. Heterogeneous graph attention network. In *WWW*, 2019b.



Reference VI

- B. Weisfeiler and A. Leman. A reduction of a graph to a canonical form and an algebra arising during this reduction (in russian). *Nauchno-Tekhnicheskaya Informatsia*, 9:12–16, 01 1968.
- Matthew Wilson and Tucker Hermans. Learning to manipulate object collections using grounded state representations. In *3rd Conference on Robot Learning*, 2019.
- Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr., Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. Simplifying graph convolutional networks. In *ICML*, 2019.
- Bingbing Xu, Huawei Shen, Qi Cao, Yunqi Qiu, and Xueqi Cheng. Graph wavelet neural network. In *ICLR*, 2019a.
- Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019b.



Reference VII

- Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.
- Liang Yao, Chengsheng Mao, and Yuan Luo. Graph convolutional networks for text classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:7370–7377, 2019. doi: 10.1609/aaai.v33i01.33017370.
- Wayne W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 33 (4):452–473, 1976.



Reference VIII

Vinicius Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David Reichert, Timothy Lillicrap, Edward Lockhart, Murray Shanahan, Victoria Langston, Razvan Pascanu, Matthew Botvinick, Oriol Vinyals, and Peter Battaglia. Relational deep reinforcement learning. *arXiv preprint arXiv:1806.01830*, 2018.

Ziwei Zhang, Peng Cui, and Wenwu Zhu. Deep learning on graphs: A survey. *arXiv preprint arXiv:1812.04202*, 2018.

Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434*, 2018.

Marinka Zitnik, Monica Agrawal, and Jure Leskovec. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(13):457–466, 2018.

