图表示学习与图神经网络

冯吕　　201928013229158
张佳锋　201918007329007

摘要

TODO

关键字：图表示学习、Network Embedding、RecGNN、ConvGNN、GAE、STGNN

1　背景

近年来，深度学习在许多机器学习任务上取得了革命性的进展，比如图像分类 [1]、目标检测 [2]、语音识别 [3] 和机器翻译 [4] 等。深度学习在这些领域的成功部分归因于快速发展的计算资源（如 GPU），大量训练数据的可用性，以及深度学习从欧几里得数据抽取特征表示的有效性。图像、文本和视频等均可表示为欧几里得空间中的数据。以图像为例，我们可以将图像表示为欧几里得空间中的规则网络，卷积神经网络（GNN）[5] 能够利用图像数据的平移不变形，局部连通性和合成性，从而提取与整个数据集共享的局部特征，以进行各种图像分析。

与此同时，图（Graph）数据易于描述对象之间的复杂关系，因此越来越来的任务用图结构来表示数据，比如社交网络、推荐系统以及知识图谱等。以社交网络为例，在社交网络图中，节点表示个人或组织，而边表示节点之间的各种关系，通过对社交网络进行挖掘，能够发现虚拟社区，进行用户行为分析等。然而，图数据的复杂性对现有的机器学习算法提出了重大挑战。由于图具有复杂的拓扑结构，没有固定的节点顺序，并且图可能是动态的，因此一些重要的操作例如卷积，在图像域中易于计算，但是应用于图域却非常困难。此外，现有机器学习算法的核心假设是实例彼此独立，但是该假设不再适用于图数据，因为每个实例（节点）通过引用、朋友关系等各种类型的链接与其它实例相关联。

为了应对图数据问题的复杂性，图表示学习应运而生。图数据本身是非欧几里得空间的数据，为了能够将传统的机器学习方法运用到图数据上，我们需

要将图数据表示成欧几里得空间中的数据，而这正是图表示学习所研究的内容。受自动编码器（Auto-encoder）[6] 和词嵌入（Word-Embedding）[7] 的启发，我们希望能够将网络中的节点表示为特征向量，使得该向量能够自带节点信息，例如在特征空间上，相似的节点会离得特别近，这将有利于机器学习的任务

在图表示学习中，给定一个图 $G(V, E, W)$，其中，V 是节点集，E 是节点之间的边集，W 是边上的权重集合，我们通过 embedding 方法将每一个节点表示成一个 d 维向量，简单来说就是将节点映射成低维向量，如图1所示。更具体地说，图表示学习由两个关键部分组成：

- 编码器（encoder）：将每一个节点映射成一个低维向量。

$$\mathrm{ENC}(v) = \mathbf{Z}_v$$

  其中，v 是图中的节点，$\mathbf{Z}_v$ 是一个 d 维向量。

- 相似度（similarity）函数：指定向量空间中的向量关系如何映射到原图中节点之间的关系。

$$\mathrm{similarity}(u, v) \approx \mathbf{z}_v^T \mathbf{z}_u$$

  其中，$\mathrm{similarity}(u, v)$ 表示 u, v 两个节点在图中的相似度，$\mathbf{z}_v^T \mathbf{z}_u$ 表示两个节点向量的点积。该公式也正是图表示学习的优化目标。

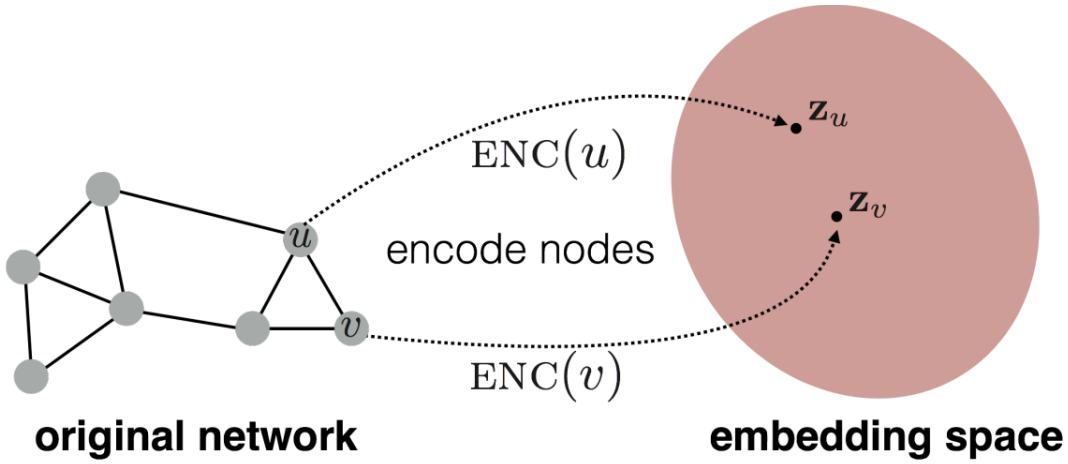

图 1: Node embedding：将节点映射到低维向量空间

最简单的编码器就是如下的 embedding 表：

$$\mathrm{ENC}(v) = \mathbf{Z}_\mathbf{v}$$

其中，$\mathbf{Z}$ 是 $\mathbb{R}^{d\times|V|}$ 维的矩阵，每一列是一个节点向量，$\mathbf{v}$ 是一个指示符向量，向量中只有一个位置为 1，其余位置全为 0，因此，每一个节点映射到一个唯一的特征向量。这一类编码方法有 *DEEPWALK* [8]、*LINE* [9] 和 *Node2vec* [10] 等，而这些方法的主要区别在于节点相似度如何定义。

然而，上面谈到的这一类浅层的节点表示学习方法存在如下一些弊端：首先，由于每一个节点被映射到一个唯一的特征向量，并且节点之间没有参数共享，因此，总要需要 $O(|V|)$ 个参数；其次，由上文的介绍容易知道，该方法仅能对训练过程中出现的节点进行编码，而无法编码训练过程中从未出现过的节点；另外，无法合并节点特征。因此，我们需要提出一些更加"Deep" 的 node embedding 方法，这就是图神经网络（GNN）。

图神经网络使用更加复杂的函数来对节点进行编码，对一个节点进行编码时通常还需要用到其邻居节点的信息。在本文中，我们沿用 Wu 等 (2019) [11] 提出的图神经网络分类方法，主要将 GNN 分为如下四类：

- 循环图神经网络（RecGNN)：RecGNN 旨在通过循环神经网络架构来学习节点表示，该网络假设图中的节点能够持续地与邻居节点交换信息，直到达到稳定平衡点。
- 卷积图神经网络（ConvGNN）：ConvGNN 将卷积操作从网格数据泛化到了图数据上，其主要思想是通过汇总节点自身的特征 $\mathbf{x}_v$ 和邻居节点的特征 $\mathbf{x}_u$ 来学习节点的表示。
- 图自动编码器（GAE）：GAE 是一种无监督学习方法，其能够将节点或图编码到特征向量空间，并从编码信息中重构图信息。
- 时空图神经网络（STGNN）：STGNN 旨在从时空图中学习隐藏模式。

图2是图表示学习的分类树状图，正如上文所谈到的，图神经网络是将深度学习方法应用到图表示学习的一种高级方法，也是本文阐述的重点。在本文的第二部分，我们将对浅层的节点表示学习包括 *DEEPWALK*、*LINE* 和 *Node2vec* 进行简要介绍；在第三部分，我们将重点介绍图2中的四种不同类别的图神经网络；最后，我们将对图表示学习和图神经网络当前的实际应用以及当前面临的困难和挑战进行简要介绍。

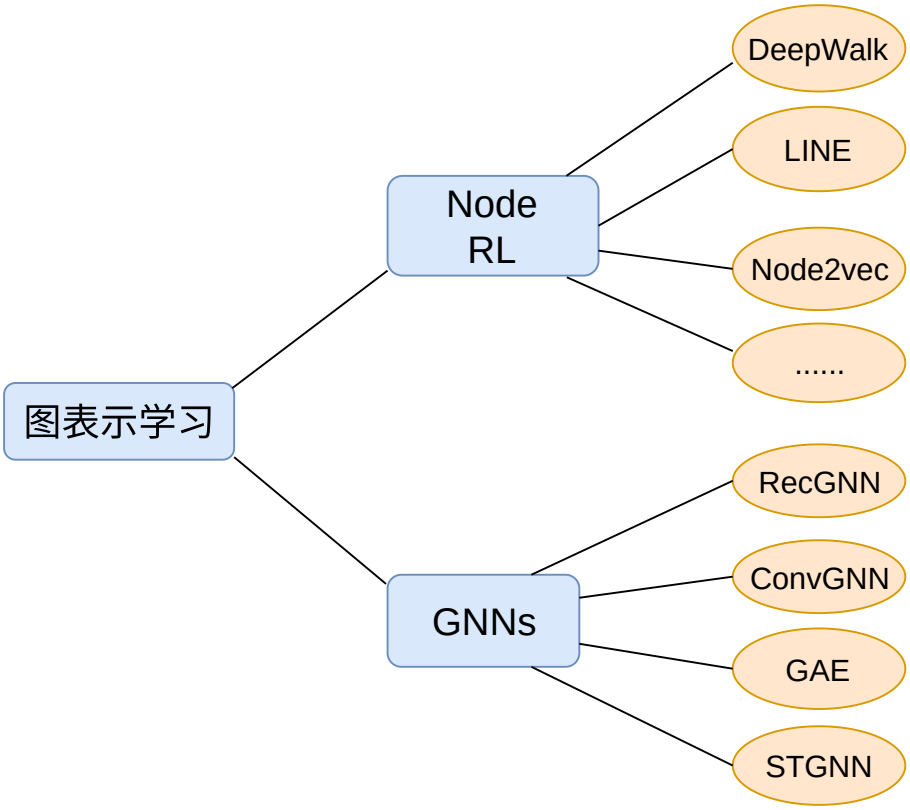

图 2: 图表示学习分类树状图

2 节点表示学习

在第1节中，我们谈到，浅层的 node embedding 是一类较为简单的图表示学习方法，而其中比较具有代表性的就是 *DEEPWALK*、*LINE* 以及 *Node2vec*，因此，在本节中，我们将简要介绍这三种 node embedding 方法。

2.1 *DEEPWALK*

DEEPWALK 是由 Perozzi 等 [8] 提出的一个非常经典的节点表示学习方法，其核心思想是将 network embedding 与自然语言处理中极为重要的 word embedding 方法 word2vec 联系起来，从而将 network embedding 问题转换成了一个 word embedding 问题。

DEEPWALK 算法由两个主要的部分组成。前一部分是随机游走生成器。*DEEPWALK* 从每一个节点出发 γ 次，每一次都采取均匀采样的方式选择当前节点的邻接节点作为下一步的节点随机游走，当游走的路径长度达到 t 之后，停止一次游走。按照这种方式便可以生成一个个节点游走序列，每个序列称为一个 *walk*，如图3所示。每个 *walk* 被当作 word2vec 中的一个句子，而每个节点即为

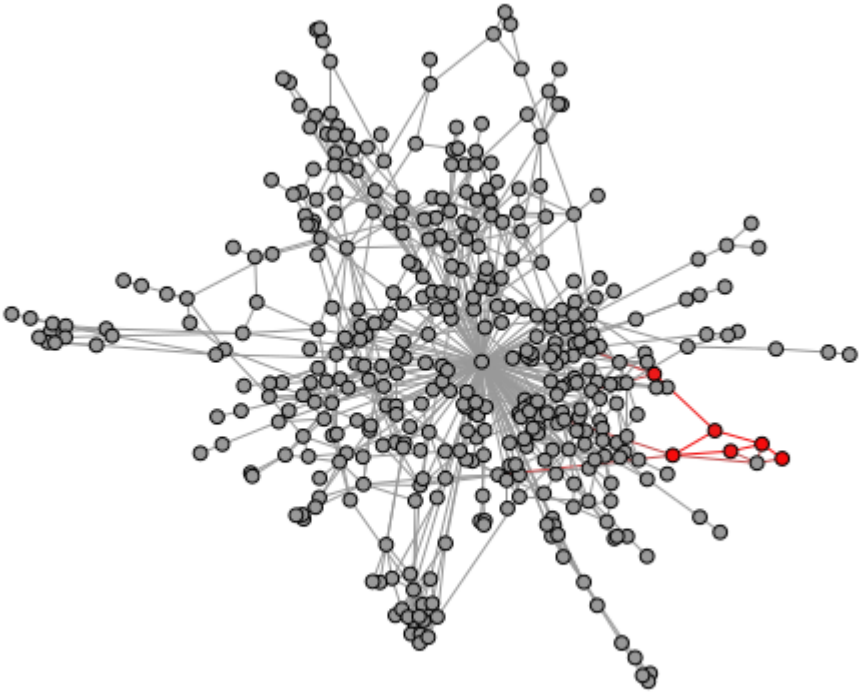

图 3: 随机游走生成示意图 [8]

word2vec 中的一个词。而算法的第二部分即为将语言模型 SkipGram 应用到随机游走生成的 *walk* 上面，使用大小为 w 的滑动窗口作为一个 walk 的 context，使用一个 context 的中心节点去推测 context 中的所有其它节点。因此，目标函数即为

$$\min_{\Phi} -\log Pr(\{v_{i-w}, \cdots, v_{i-1}, v_{i+1}, \cdots, v_{i+w}\} \mid \Phi(v_i))$$

其中，Φ 为映射函数 $\Phi : v \in V \mapsto \mathbb{R}^{|V| \times d}$

在实验环节，*DEEPWALK* 选择多标签分类作为评价算法性能的指标。评价中将通过 *DEEPWALK* 学习获得的 embedding 按照不同的比例划分为训练集和测试集，训练集作为 N 个 *one-vs-rest* 对率回归分类器的训练数据，将其中置信度最高的 k 个类别作为节点的预测类别。其中，N 为类别的个数，k 为节点的表签数。

2.2 *LINE*

DEEPWALK 通过随机游走的方式，将图结构数据转化为了自然语言处理的任务来完成。然而，在图结构数据中，节点之间的关系比词上下文的关系更为

复杂；而且，图结构数据中的边通常具有权重，使用现有的 word2vec 方法无法很好的应对这个问题；另外，在真实数据中，图的规模通常过于庞大，存下所有的 *walk* 将会带来巨大的开销。

Tang 等 [9] 提出了一种新的 network embedding 方法：*LINE*。该算法适用于各种类型的图，如有向图、无向图、有权图和无权图，并且能够支持百万节点的图。而且，该算法能够同时保留节点的一阶相似度和二阶相似度。

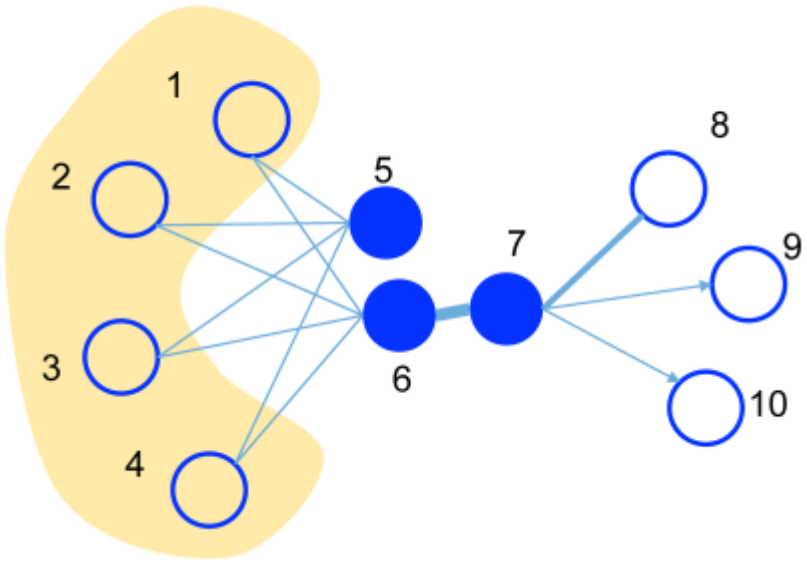

图 4: 一阶相似度和二阶相似度示意图 [9]

一阶相似度是两个顶点之间的局部成对相似度。对于由边 (u, v) 相连的一对节点来说，边的权重 w_{uv} 即为 u 和 v 的一阶相似度，如果两个节点之间没有边相连，那么一阶相似度即为 0。换句话说，一阶相似度就是两个节点直接联系的紧密程度，保留一阶相似度就是保留节点之间的边权。例如，图4中的节点 6 和 7 之间有一条权重很大的边相连，因此这两个节点之间具有非常大的一阶相似度，所以当映射到低维空间之后，这两个节点应该离得很近。

二阶相似度是一对节点的邻居网络结构之间的相似度。从数学上来说，令 $p_u = (w_{u,1}, \cdots, w_{u,|V|})$ 表示节点 u 和其它所有节点的一阶相似度，则节点 u 和 v 的二阶相似度即为 p_u 和 p_v 的相似度。如果没有任何节点同时与 u 和 v 相连，那么它们之间的二阶相似度即为 0。例如，图4中的节点 5 和 6 尽管没有边相连，但是它们有许多公共邻居，也就是具有很高的二阶相似度，因此这两个节点的低维表示也应该离得很近。

在 *LINE* 中，两个节点实际的一阶相似度定义如下

$$\hat{p}_1(i,j) = \frac{w_{ij}}{W}$$

其中，W 是所有边权重之和。而两个节点 embedding 的相似度定义为

$$p_1(v_i, v_j) = \frac{1}{1 + \exp(-\vec{u}_i^T \cdot \vec{u}_j)}$$

其中，$\vec{u}_i \in R^d$ 是节点 v_i 在低维向量空间的 embedding。目标函数设为实际相似度与表示相似度之间的 KL 散度，这样一来，只要最小化 KL 散度 (下式中约去了一些常数)，就能保证表示相似度尽量接近实际相似度：

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

需要注意的是，一阶相似度仅适用于无向图，通过最小化上述目标函数，就可以找到每一个节点在 d 维空间的表示。

二阶相似度既适用于无向图，也适用于有向图。两个节点实际的二阶相似度定义如下

$$\hat{p}_2(v_j|v_i) = \frac{w_{ij}}{d_i}$$

其中，w_{ij} 是边 (v_i, v_j) 的权重，d_i 是节点 v_i 的出度。两个节点的 embedding 相似度定义为

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

其中，V 为节点 v_i 的所有邻居。与一阶相似度不同的是，对于邻居节点，使用了另一组 embedding，称为 context。目标函数依旧为 KL 散度：

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

最终要获得同时包含有一阶相似度和二阶相似度的 embedding，只需要将通过一阶相似度获得的 embedding 与通过二阶相似度获得的 embedding 拼接即可。

实验证明，当考虑了一阶相似度之后，*LINE* 在节点分类任务中的效果要好于 *DEEPWALK*。

2.3 *Node2vec*

Node2vec [10] 是一份基于 *DEEPWALK* 的延伸工作,它改进了 *DEEPWALK* 随机游走的策略。*DEEPWALK* 根据边的权重进行随机游走，而 *Node2vec* 加了一个权重调整参数 α，最终生成的随机序列是 DFS 和 BFS 的结合。随机游走产生的序列，仍然使用 SkipGram 模型进行训练。

Node2vec 认为，现有的方法无法很好的保留网络的结构信息，例如图5所示，有一些点之间的连接非常紧密 (比如 $u, s1, s2, s3, s4$)，他们之间就组成了一个社区 (community)；同时，网络中可能存在着各种各样的社区，而有的结点在各个社区中可能又扮演着相似的角色 (比如 u 与 $s6$)。因此，*Node2vec* 的优化目标就是让同一个社区内的结点表示能够相互接近，并且在不同社区内扮演相似角色的结点表示也要相互接近。

Node2vec 同时结合了深入优先搜索（DFS）和广度优先搜索（BFS）来实现上述优化目标。如图5所示，深度优先游走策略将会限制游走序列中出现重复的结点，防止游走掉头，促进游走向更远的地方进行。而广度优先游走策略相反将会促进游走不断的回头，去访问上一步结点的其他邻居结点。这样一来，当使用广度优先策略时，游走将会在一个社区内长时间停留，使得一个社区内的结点互相成为 context，这也就达到了第一条优化目标。反之，当使用深度优先的策略的时候，游走很难在同一个社区内停留，也就达到了第二条优化目标。

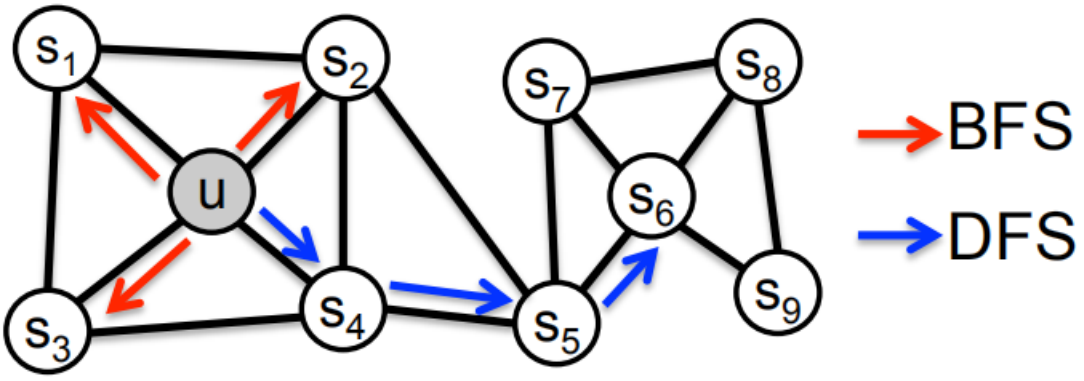

图 5: BFS 和 DFS 从节点 u 开始的搜索策略 [10]

如图6所示，在一次随机游走过程中，假设当前游走路径从 t 到达 v（当前位于 v），当计算从 v 到下一个节点 x 的转移概率时，需要先计算搜索偏差因子

α，其定义为

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

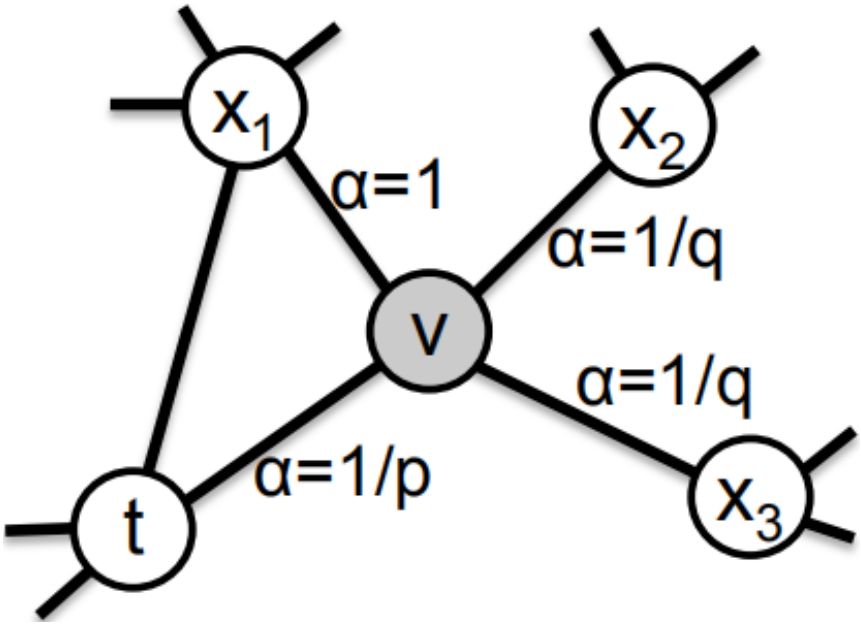

图 6: 在 node2vec 中随机游走的说明 [10]

其中，d_{tx} 表示 t 与 x 之间的最短路径，而参数 p, q 用于控制深度优先搜索策略和广度优先搜索策略的比重。当计算出偏差因子之后，定义 v 和 x 之间的非归一化转移概率为

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

其中，w_{vx} 表示边 (v,x) 的权重。最后，节点 v 到 x 的转移概率为

$$P(c_i = x | c_{i-1} = v) = \begin{cases} \frac{\pi_{vx}}{Z} & \text{if } (u,v) \in E \\ 0 & \text{if } (u,v) \notin E \end{cases}$$

Z 为归一化因子。值得一提的是，当 $p = q = 1$ 时，*Node2vec* 退化为 *DEEPWALK*。

实验证明，当调整好适当的 p, q 值之后，*Node2vec* 在多标签分类任务中的效果要好于 *DEEPWALK*。

3 图神经网络

3.1 循环图神经网络

3.2 图卷积神经网络

卷积神经网络 (CNN) 无法处理非欧几里得 (Non Euclidean Structure) 的数据，学术上的表达是传统的离散卷积在 Non Euclidean Structure 的数据上无法保持平移不变性。也就是说在拓扑图中每个顶点的相邻顶点数目都可能不同，那么也就无法用一个同样的尺寸的卷积核来进行卷积运算。CNN 无法处理 Non Euclidean Structure 的数据，又希望在拓扑图上有效的提取空间特征来进行学习，所以 GCN 成为研究重点。图结构的数据在现实应用中非常常见，比如: Social Networks, Citation Netwoks, Knowledge Graphs 等。图半监督学习的设定是，在给定的图结构的数据中，只有少部分节点是有标记的，大部分节点是未标记的。其任务就是预测出未标记节点的 label。

经典的方法大概可以分为两类：Standard Approach 和 Embedding-based Approach。来自阿姆斯特丹大学的作者 Thomas N.Kipf Max Welling 采用图卷积神经网络来进行半监督分类 [12]。接下来，将对图卷积神经网络进行简要的介绍。

图卷积神经网络 (Graph Convolutional Networks)，对于一个图 $G = (V, E)$，我们有输入 X 是一个 $N \times D$ 的矩阵，表示每个节点的特征，同时有图的邻接矩阵 A。我们希望得到一个 $N \times F$ 的特征矩阵 Z，表示我们学得的每个节点的特征表示，其中 F 是我们希望得到的维度。

对于 L 层的神经网络来说，可以表示为 $H^{l+1} = f(H^l, A)$，其中 $H^0 = X, H^L = Z$。

传统的图像上的卷积，可以看作多余每一个节点，都加上其邻居节点的信息。而对于图来说，也可以通过 $f(H^l, A) = \delta(AH^lW^l)$ 完成类似的操作。从公式中不难看出，乘以邻接矩阵 A 就相当于对于每个节点，都加上其邻居节点的信息，这也可以看作是传统卷积的一种拓展。值得注意的是，这样的操作并没有加上自身的信息，除非这个节点有自环。因此，为了保证每个节点都能加上自己的信息，都强行加上自环。即：$\widetilde{A} = A + I$。与此同时，为了计算的简便，需要对 A 矩阵进行正则化操作，即：$D^{-\frac{1}{2}}AD^{\frac{1}{2}}$。

经过对于每个节点加上自环和正则化后，传播规则变为

$$f(H^l, A) = \delta(\widehat{D}^{-\frac{1}{2}}\widehat{A}\widehat{D}^{\frac{1}{2}}H^l W^l), \quad \widehat{A} = A + I$$

一个信号 x 与一个滤波器 $g\theta^{'}$ 的卷积定义为

$$g\theta^{'} * x \approx \sum_{k=0}^{K} \theta_k^{'} T_k(\widehat{L})x,$$

其中，$\widehat{L} = \frac{2}{\lambda_{max}}L - I_N$，$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x), T_0(x) = 1, T_1(x) = x$

在线性图卷积网络中，我们进一步的约束 $\lambda_{max} \approx 2$，在这个约束下，图卷积被简化为下式

$$g\theta^{'} * x \approx \theta_0^{'}x + \theta_1^{'}(L - I_N)x = \theta_0^{'}x + \theta_1^{'}D^{-\frac{1}{2}}AD^{\frac{1}{2}}$$

3.3 图自动编码器

3.4 时空图神经网络

4 应用

5 困难与挑战

参考文献

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[3] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, B. Kingsbury *et al.*, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal processing magazine*, vol. 29, 2012.

[4] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[6] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[7] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[8] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 701–710.

[9] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proceedings of the 24th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 2015, pp. 1067–1077.

[10] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2016, pp. 855–864.

[11] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *arXiv preprint arXiv:1901.00596*, 2019.

[12] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.