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。

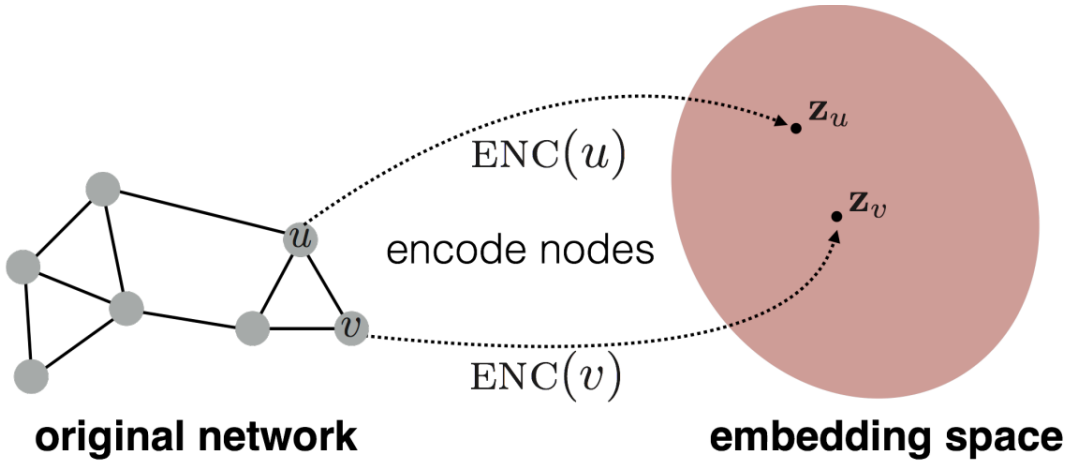


图 1: Node embedding：将节点映射到低维向量空间

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。

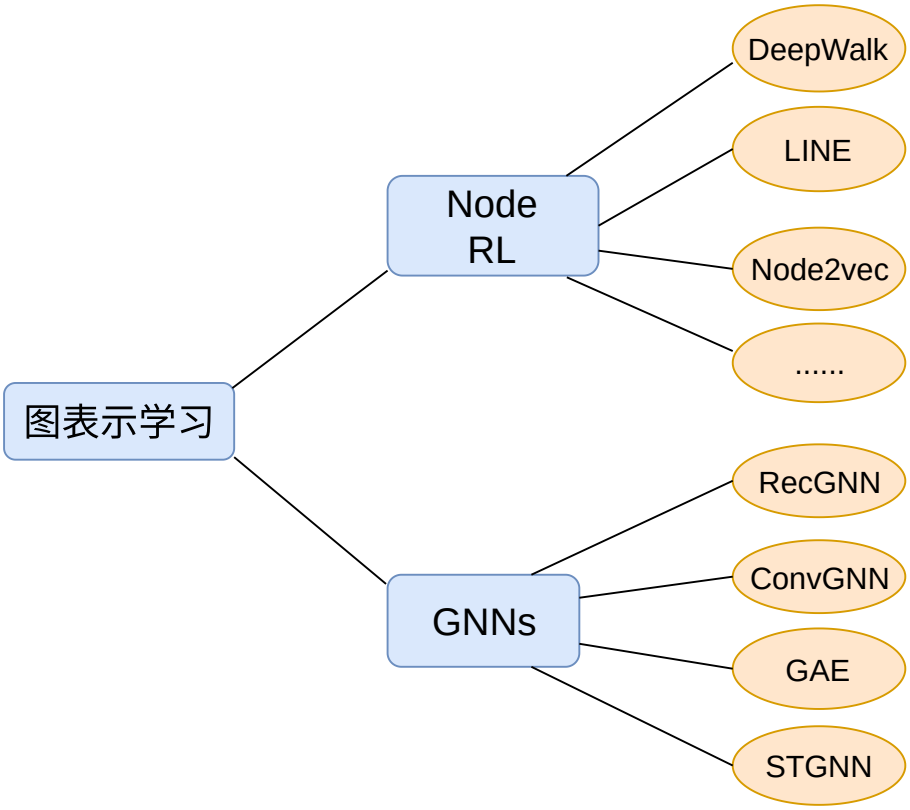


图 2: 图表示学习分类树状图

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

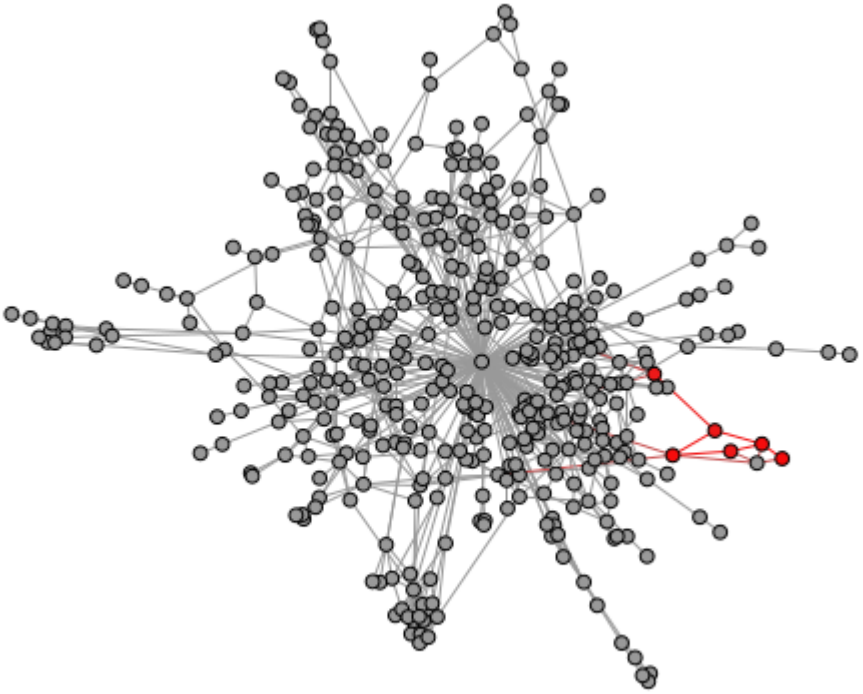


图 3: 随机游走生成示意图 [8]

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。

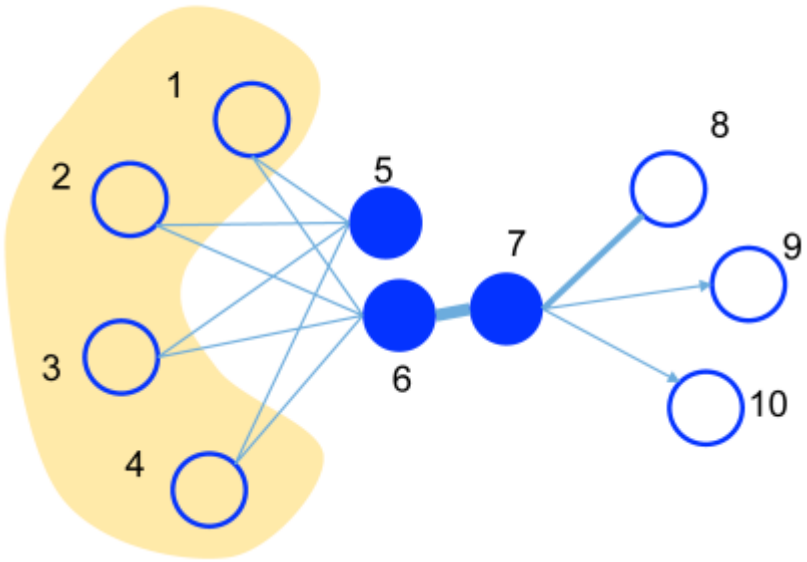


图 4: 一阶相似度和二阶相似度示意图 [9]

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。

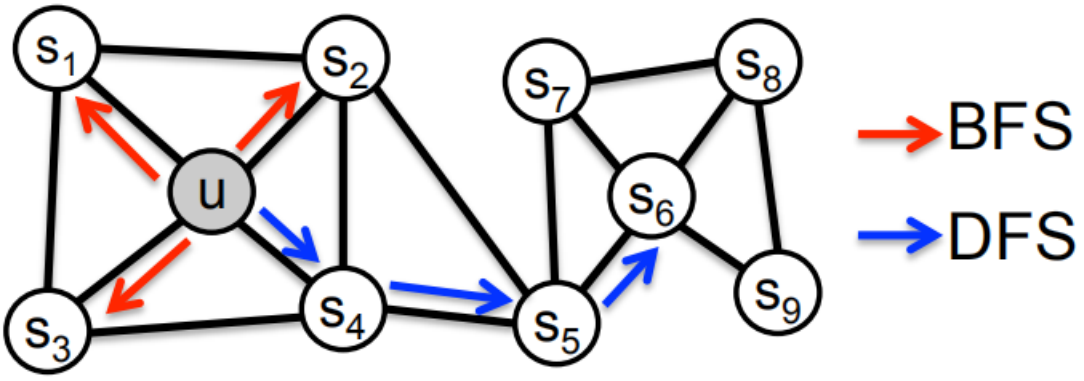


图 5: BFS 和 DFS 从节点 u 开始的搜索策略 [10]

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

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

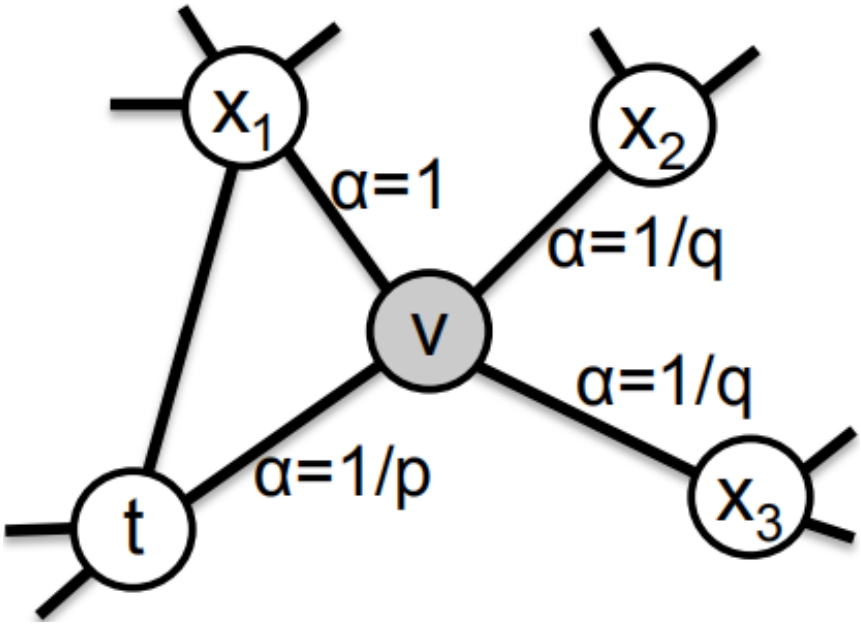


图 6: 在 node2vec 中随机游走的说明 [10]

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

循环图神经网络 (RecGNNs) 是 GNN 的先驱作品。它们在图中的节点上循环应用相同的参数集，以提取高级节点表示。受计算能力的限制，早期的研究主要集中在有向无环图。

最早由 Scarselli 等人提出的图神经网络 (GNN^*) [12]，扩展先前的递归模型以处理一般类型的图。例如，非循环，循环，定向和无向图。基于信息扩散机制，GNN^* 通过周期性地交换邻域信息来更新节点的状态，直到达到稳定的均衡。节点的隐藏状态通过以下方式反复更新。

$$h_v^{(t)} = \sum_{u \in N(v)} f(x_v, x_{v,u}^e, x_u, h_u^{t-1})$$

其中 f 是参数的函数，h_v^0 是随机初始化的。求和操作使 GNN^* 适用于所有节点，即使邻居的数量不同并且没有邻域排序。与此同时，为了确保收敛，循环函数必须是一个收缩映射，它会缩小映射后两点之间的距离。在是神经网络的情况下，必须对 $Jacobian$ 参数矩阵施加惩罚项。当满足收敛标准时，最后一步节点隐藏状态被转发到读取层。GNN^* 交替节点状态传播的阶段和参数梯度计算的阶段以最小化训练目标。该策略使 GNN^* 能够处理循环图。

图形回声状态网络 (GraphESN) 扩展了回声状态网络以提高效率。GraphESN 由编码器和输出层组成。编码器随机初始化因此无需训练。它实现了一个收缩状态转换函数，以反复更新节点状态，直到全局图状态达到收敛。然后，通过将固定节点状态作为输入来训练输出层。

门控图神经网络 (GGNN) 采用门控循环单元 (GRU) 作为循环函数，将循环减少到固定的次数。其优点是不再需要约束参数来确保收敛。这可以极大的方便训练。

节点的隐藏状态通过其先前的隐藏状态以及相邻的隐藏状态进行更新，更新公式定义如下。

$$h_v^{(t)} = GRU(h_u^{t-1}, \sum_{u \in N(v)} W h_u^{(t)})$$

其中，$h_v^{(0)} = x_v$, 与前两种方法的不同点在于，GGNN 使用反向传播的算法来学习模型的参数。对于比较大的图来说，可能会有一些麻烦。因为 GGNN 需要在

每个节点上使用多次循环函数，并将所有的中间状态都进行存储。所以 GGNN 不太适合于训练较大的网络。

随机稳态嵌入 (SSE) 提出了一种学习算法，该算法对大型图更具可稳定性。SSE 以随机和异步的方式周期性地更新节点的隐藏状态。它对一批用于状态更新的节点和一批用于梯度计算的节点进行采样。为了保持稳定性，SSE 的递归函数定义为历史状态和新状态的加权平均值，其形式为

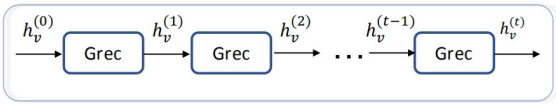
$$h_v^{(t)} = (1-\alpha)h_u^{t-1} + \alpha W_1 \delta(W_2[x_v, \sum_{u \in N(v)} [h_u^{(t-1)}, x_u]]),$$

其中，α 是超参数，$h_v^{(0)}$ 是随机初始化的。

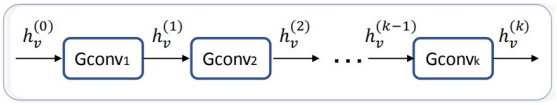
3.2 图卷积神经网络

卷积神经网络 (CNN) 无法处理非欧几里得 (Non Euclidean Structure) 的数据，即传统的离散卷积在 Non Euclidean Structure 的数据上无法保持平移不变性。因为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。

卷积图神经网络 (ConvGNN) 与循环图神经网络 (RecGNN) 密切相关。ConvGNN 不是使用收缩约束来迭代节点状态，而是在结构上使用固定数量的层 (在每一层中具有不同的权重) 来解决循环的相互依赖性。图7说明了这一主要区别。由于图卷积更加有效且易于与其他神经网络进行结合，因此，ConvGNN 近年来得到了迅速的发展。ConvGNN 分为两类，基于频谱的 ConvGNN 和基于空间的 ConvGNN。基于频谱的 ConvGNN 通过从图信号处理的角度引入滤波器来定义图卷积，其中图卷积运算被解释为消除图信号中的噪声。基于空间的 ConvGNN 通过信息聚合，以定义图卷积。自从 GCN 弥合了基于频谱的方法与基于空间的方法之间的差距以来，基于空间的方法由于其引人注目的效率，灵活性和通用性而迅速发展。



(a) 循环图神经网络 (RecGNNs)。RecGNN 在更新节点表示中使用相同的图复现层 (Grec)。



(b) 卷积图神经网络 (ConvGNN)。ConvGNN 在更新节点表示时使用不同的图卷积层 (Gconv)

图 7: RecGNNs and ConvGNNs

3.2.1 基于频谱的 ConvGNNs

基于频谱的方法在图形信号处理中具有坚实的数学基础，一般假设是无向图。无向图的数学表示是归一化图拉普拉斯矩阵，定义为

$$L = I_n - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$$

其中，D 是节点的对角矩阵。归一化图拉普拉斯矩阵具有实对称半正定的性质。利用该属性，归一化的拉普拉斯矩阵可以被分解为

$$L = U \sum U^T$$

输入信号 x 和滤波器 $g \in R^N$ 的图形卷积定义为

$$x *_G g = \mathcal{F}^{-1}(\mathcal{F}(x) \bigodot \mathcal{F}(g))$$

其中，$\bigodot$ 表示 Hadamard 矩阵乘积。如果，$g_\theta = diag(U^T g)$，卷积就会简化为

$$x *_G g_\theta = U g_\theta U^T x$$

基于频谱的 ConvGNNs 都遵循这个定义。

频谱卷积神经网络，假设滤波器是一组可学习的参数，并考虑了具有多个通道的图形信号。首先，对图的任何扰动都会导致本征基的变化。其次，学习的过滤器是域相关的，这意味着它们不能应用于具有不同结构的图。第三，特征分解需要 $O(n^3)$ 复杂度。在后续工作中，ChebNet 和 GCN 通过进行一些近似和简化将计算复杂度降低到 $O(n)$。

Chebyshev 谱 CNN(ChebNet) 通过特征值对角矩阵的 Chebyshev 多项式逼

近滤波器 g_θ，学习的权重可以在图表中的不同位置共享。ChebNet 的频谱线性映射到 [-1,1]。

CayleyNet 进一步应用 Cayley 多项式，它是参数有理复合函数，用于捕获窄频带。CayleyNet 的频谱图卷积定义为

$$x *_G g_\theta = c_0 x + 2Re\{\sum_{j=1}^{r} c_j (hL - iI)^j (hL + iI)^{-j} x)\}$$

其中，Re 返回复数的实部，c_0 是一个实数系数，c_j 是一个复数系数，i 是虚数，h 是控制 Cayley 滤波器频谱的参数。在保留空间局部性的同时，CayleyNet 显示 ChebNet 可以被视为 CayleyNet 的特例。

最近的几项工作通过探索替代对称矩阵对 GCN 进行了改进。自适应图形卷积网络 (AGCN) 学习由图形邻接矩阵未指定的隐藏结构关系。它通过可学习的距离函数构造所谓的残差图邻接矩阵，该函数将两个节点的特征作为输入。

3.2.2 基于空间的 ConvGNNs

由于传统 CNN 在图像上的卷积运算，基于空间的方法是根据节点的空间关系来定义图形卷积。图像可以被视为图形的特殊形式，每个像素代表一个节点，每个像素直接连接到其附近的像素，如8(a) 所示。使用 3×3 窗口，每个节点的邻域是其周围的八个像素。然后通过获取每个通道上的中心节点及其邻居的像素值的加权平均值，将滤波器应用于该区域。类似地，基于空间的图卷积将中心节点的表示与其邻居的表示进行卷积，以导出中心节点的更新表示，如图8(b) 所示。从另一个角度来看，基于空间的 ConvGNN 与 RecGNN 共享信息传播/消息传递是相同概念。空间图卷积运算实质上沿边传播节点信息。

(a) 2 维卷积 (b) 图卷积

图 8: 2D 卷积和图卷积

图形神经网络 (NN4G) 是基于空间的 ConvGNN 的第一项工作。NN4G 通过直接汇总节点的邻域信息来执行图卷积。它还会应用剩余连接和跳过连接以

存储每个层上的信息。结果，NN4G 通过以下方式导出其下一层节点状态

$$h_v^{(k)} = f(x_v W^{(k-1)} + \sum_{k-1}^{i=1} \sum_{u \in N(v)} h_u^{(k-1)} \theta^{(k-1)})$$

其中，f 是激活函数，初始 $h_v^{(0)} = 0$

它类似于 GCN 的形式。主要区别在于 NN4G 使用非标准化邻接矩阵，这可能会导致数值不稳定性问题。语境图马尔可夫模型 (CGMM) 提出了一种基于 NN4G 的概率模型。在保持空间局部性的同时，CGMM 具有概率可解释性的好处。

消息传递神经网络 (MPNN) 概述了基于空间的 ConvGNN 的一般框架。它将图形卷积视为消息传递过程，其中信息可以直接沿着边缘从一个节点传递到另一个节点。MPNN 运行 K 步消息传递迭代以使信息进一步传播。信息传递函数，即空间图卷积定义如下，

$$h_v^{(k)} = U_k(h_v^{k-1}, \sum_{u \in N(v)} M_k(h_v^{(k-1)}, h_u^{(k-1)}, x_{uv}^e))$$

其中 $h_v^0 = x_v, U_k, M_k$ 是通过学习得到的。由于节点的邻居数量可以从一千到甚至更多，因此无法获取节点邻域的全部大小。GraphSage 采用抽样来获得每个节点固定数量的邻居。

图注意力网络 (GAT) 假设相邻节点对中心节点的贡献既不像 GraphSage 那样相同，也不像 GCN 那样预先确定。GAT 采用注意机制来学习两个连接节点之间的相对权重。GAT 的图卷积操作定义为，

$$h_v^{(k)} = \delta(\sum_{u \in \mathcal{N}(\sqsubseteq) \cup v} \alpha_{vu} W_k^{(k-1)}, h_u^{(k-1)})$$

其中，$h_v^0 = x_v$, 注意力权重 α_{uv} 控制节点 u 到 v 的连接权重。计算方法如下，

$$\alpha_{uv} = softmax(g(a^T[W^{(k-1)}h_v || W^{(k-1)}h_u]))$$

其中，g 是一个 LeakRELU 激活函数，a 是一个可以学到的参数向量。softmax 函数保证了在 v 所有的邻居节点的注意力权重总和到 1。

虽然 GAT 假设注意力头的贡献相等，但门控注意网络 (GAAN) 引入了一种自我关注机制，该机制计算每个注意力头的额外注意力得分。除了在空间上

应用图注意外，GeniePath 进一步提出了一种类似 LSTM 的门控机制来控制图卷积层的信息流。

训练效率方面的改进通常需要训练 ConvGNN 将整个图形数据和所有节点中间状态保存到内存中。ConvGNN 的全批次训练算法明显受到内存溢出问题的困扰，尤其是当一个图包含数百万个节点时。为了节省内存，GraphSage 提出了一种针对 ConvGNN 的批量训练算法。它通过以固定的样本大小按 K 步递归扩展根节点的邻域来对植根于每个节点的树进行采样。对于每棵采样树，GraphSage 通过从下到上分层汇总隐藏节点表示来计算根节点的隐藏表示。

基于频谱和空间模型之间的比较：频谱模型在图形信号处理中具有理论基础。通过设计新的图形信号滤波器，理论上可以构建新的 ConvGNN。但是，由于效率，通用性和灵活性问题，与频谱模型相比，空间模型更为可取。首先，频谱模型的效率不如空间模型。频谱模型要么需要执行特征向量计算，要么需要同时处理整个图形。空间模型对大型图更具可伸缩性，因为它们通过信息聚合在图域中直接执行卷积。可以在一批节点而不是整个图中执行计算。其次，依赖于图傅立叶基础的频谱模型不能很好地推广到新图。他们假设一个固定的图。对图的任何扰动都会导致本征基的变化。另一方面，基于空间的模型在每个权重可以在其上的节点上本地执行图卷积，在不同的位置和结构之间轻松共享。第三，基于频谱的模型仅限于在无向图上运行。基于空间的模型更灵活地处理多源图输入，例如边输入，有向图，有符号图和异构图，因为这些图输入可以轻松地合并到聚合函数中。

3.2.3 图池化模型

在 GNN 生成节点功能之后，我们可以将它们用于最终任务。但是直接使用所有这些特征在计算上具有挑战性，因此需要采用下采样策略。根据目标及其在网络中扮演的角色，该策略给出了不同的名称：(1) 池化操作旨在通过对节点进行下采样来减小参数的大小，以生成更小的表示，从而避免排列不变性和计算复杂性问题; (2) 读出操作主要用于基于节点表示生成图级表示。他们的机制非常相似。在本章中，我们使用池化来指代应用于 GNN 的各种下采样策略。

在一些较早的工作中，图粗化算法使用特征分解来基于图的拓扑结构来粗化图。但是，这些方法存在时间复杂性问题。Graclus 算法是特征分解的替代方法，用于计算原始图的聚类版本。最近的一些工作将其作为池操作来粗化图。

如今，平均值/最大值/总和池是实现下采样的最原始有效的方法，因为计算

池化窗口中的均值/最大值/总和值很快：

$$h_G = mean/max/sum(h_1^k, h_2^k, ..., h_n^k)$$

其中 K 是最后一个图卷积层的索引。Henaff 等人的研究 [13] 表明，在网络开始时执行简单的最大/均值池对于降低图域中的维数并降低图傅里叶变换操作的成本尤为重要。

总体而言，池化是减少图形大小的基本操作。然而如何提高汇集的有效性和计算复杂性仍然是一个悬而未决的问题。

3.3 图自动编码器

3.4 时空图神经网络

在现实世界的许多应用中，图形无论是在图形结构还是在图形输入方面都是动态的。时空图神经网络 (STGNNs) 在捕捉图的动态性方面占有重要地位。这类方法的目的是在假设连接节点之间相互依赖的情况下，对动态节点输入进行建模。例如，交通网络中节点由放置在道路上的速度传感器组成，边权重由传感器对之间的距离决定。由于一条道路的交通状况可能取决于其相邻道路的状况，因此在进行交通速度预测时必须考虑空间相关性。作为一种解决方案，STGNNs 同时捕获图的空间和时间依赖关系。STGNNs 的任务可以是预测未来的节点值或标签，或者预测时空图标签。STGNNs 遵循两个方向：基于 RNN 的方法和基于 CNN 的方法。

基于 RNN 的方法：基于 RNN 的方法通过使用图形卷积过滤输入和隐藏状态来捕获时空依赖性。为了说明这一点，假设一个简单的 RNN 采用这种形式

$$H^{(t)} = \delta(WX^{(t)} + UH^{(t)} + b)$$

其中，$X^{(t)} \in R^{n \times d}$ 是时间 t 步的节点特征矩阵，插入图卷积后，上式变为

$$H^{(t)} = \delta(Gconv(X^{(t)}, A; W) + Gconv(H^{(t-1)}, A; U) + b)$$

其中，Gconv 为图卷积层。

Structural-RNN [14] 提出了一个递归框架来预测每个时间步的节点标签。它包括两种 RNN，即节点 RNN 和边缘 RNN。每个节点和每个边的时间信息分别

通过节点-RNN 和边缘-RNN。为了合并空间信息，节点 RNN 将边-RNN 的输出作为输入。由于为不同的节点和边缘假定不同的 RNN 会显着增加模型的复杂性，因此它将节点和边缘分成语义组。同一语义组中的节点或边共享相同的 RNN 模型，从而节省了计算成本。基于 RNN 的方法会有耗时的迭代和梯度爆炸/消失问题，特别是在结合 ConvGNN 时比较显著。

作为替代的方法，基于 CNN 的方法通过非递归的几乎等效的计算，稳定的梯度和较低的存储需要来解决时空图问题。

基于 CNN 的方法：通过一个一维的图卷积层来分别学习时间和空间的依赖关系。假设一个图时空网络的输入是张量 $\mathcal{X} \in R^{T\times n\times d}$, 在图卷积层在 $\mathcal{X}_{[i,:,:]}$ 收集信息时，一维的图卷积层沿着时间轴，依次滑过 $\mathcal{X}_{[:\,i,:]}$ 来为每个节点收集信息。CGCN 使一维卷积层通过 CheNet 或者 GCN 一体化，通过叠加按顺序排列的一个门控一维卷积层，一个图卷积层和另外一个门控一维卷积层来构建一个时间块。ST-GCN 使用一个一维卷积层和一个 PGC 层组成一个时间块。前面的方法都是使用预定义的图结构，他们假设预定义的图结构反应了节点间真正的依赖关系。然而，由于在时空环境中有许多图形数据快照，因此可以从数据中自动学习潜在的静态图形结构。为了实现这一目的，Graph WaveNet [15] 提出了一种自适应邻接矩阵来执行图卷积。自适应邻接矩阵定义为

$$A_{adp} = SoftMax(ReLU(E_1 E_2^T))$$

其中 E1 表示源节点嵌入，E2 表示嵌入可学习参数的目标节点。通过将 E1 与 E2 相乘，可以获得从源节点到目标节点的依赖性权重。利用复杂的基于 CNN 的时空神经网络，Graph WaveNet 在没有给出邻接矩阵的情况下表现良好。

通过学习隐藏的静态空间依赖性的意义在于，可以帮助研究人员发现网络中不同实体之间的可解释和稳定的相关性。然而，在某些情况下，学习潜在的动态空间依赖性可以进一步提高模型精度。例如，在交通网络中，两条道路之间的行程时间可能取决于它们当前的交通条件。GaAN [48] 采用注意机制通过基于 RNN 的方法学习动态空间依赖性。该功能用于在给定其当前节点输入的情况下更新两个连接节点之间的边缘权重。ASTGCN 采纳了空间注意功能和时间注意功能，以通过基于 CNN 的方法来学习潜在的动态空间依赖性和时间依赖性。学习潜在空间依赖性的共同缺点是它需要计算每对节点之间的空间依赖性权重，其时间复杂度为 $O(n^2)$

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] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2008.

[13] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," *arXiv preprint arXiv:1506.05163*, 2015.

[14] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-rnn: Deep learning on spatio-temporal graphs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5308–5317.

[15] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph wavenet for deep spatial-temporal graph modeling," *arXiv preprint arXiv:1906.00121*, 2019.