GNN学习笔记

I.前置铺垫:

GNN的发展缘由:传统机器学习在处理欧几里得空间的数据已经得到了很好的发展,但是对于对于存在更加普遍的非欧几里得空间的数据时,特别是对于图这种典型的非欧几里得数据,由于节点之间**存在复杂关系,以及内在联系**.因此传统的机器学习算法并不能很好处理这种类型的数据.

GNN的分类:

> - 循环图神经网络(recurrent graph neural network)
> - 卷积图神经网络(convolution graph neural network)
> - 自编码器(Autoencoder)
> - 时空图神经网络(spatial-temporal graph neural network)

II.面临的挑战以及部分方法

图神经网络面临的挑战:

挑战一:独立实体假设失效

现象: 传统机器学习的前提假设之一:独立实体假设失效(即数据不满足独立同分布)
原因: 图数据节点之间具有强数据关联以及复杂关系,实质上边本身就是一种数据强关联的表示.

挑战二:图类型多样

现象: 图结构数据包含内容多样,节点和边的表示方式多样,导致目前机器学习方式无法很好兼容以及迁移到图神经网络中
原因: 图大致上可以分为同构图以及异构图,异构图中不仅节点可能存在异构,边也可能存在异构现象,因此对于在图上的操作通常情况下无法很好的兼容所有情况.

挑战三:数据不规整

现象: 对于一些较为成熟且重要的数据处理方式(如卷积操作),无法简单迁移到图神经网络中(由于图邻居节点个数不确定),需要进行拓展或者重新定义
原因: 与欧几里得空间数据不同,非欧几里得空间的数据通常不具有很好的对称以及结构化的表示方式,我们无法简单表示邻居节点,或者邻居节点的个数通常不能确定.

GNN方法引入情况(部分):

> - Bronstein等人使用几何语言描述非欧空间的机器学习
> - Hamilton等人最早提出使用图嵌入的方式来解决gnn
> - Battaglia等人提出图网络时解决相关数据的关键(building blocks)
> - Lee等人在图神经网络中尝试使用不同的注意力机制

III.背景以及相关定义

背景:

下面主要按照时间先进行描述:

> - 1997年,Sperduti等人在有向无环图(DAG)中第一次使用神经网络的方式进行训练[13],该做法启发了后续针对GNN的研究
> - 2005年,Gori等人提出正式提出图神经网络的问题[14],之后Scarselli(2009)以及Gallicchio(2010)对图神经网络进行进一步的阐述[15].
> - 早期对GNN的研究主要针对于**循环图神经网络**(RecGNNs),其基本思想在于在传递邻居信息时,通过不断迭代直到稳态,以此来学习目标的节点表示,这样的作法通常情况下会带来大量的计算开销,但是仍有不少尝试以及优化[17][18]
> - 由于早期CNN在计算机视觉领域获得的成功,因此该思想也被引入至GNN领域,但是对于卷积方式,主要有两种流派:基于频域的卷积(基于信号处理)以及基于空域的卷积(基于节点)
> - 基于频域的卷积方式在2013年获得了较为突出的成果[13],Bruna等人使用频域方法,将卷积操作定义为傅里叶频域计算图拉普拉斯(graph Laplacian)的特征值分解,在此之上定义卷积核以及滤波器进行特征提取.之后针对其有人进一步进行拓展提升以及应用,相关资料可以参考[20][21][22][23]
> - 基于空域的卷积在2009年由Micheli提出,但是其做出的贡献实际上并不显著,而之后更多的基于空域的卷积图神经网络陆续被提出[25][26][27]
> - 而再之后近几年图自编码(GAE)以及时空图神经网络陆续被提出.

相关概念区分:

图神经网络(GNN)与图嵌入(network embedding)

图嵌入主要将图节点,进行**低维向量转化**,并且**保留图节点信息以及网络拓扑关系**,对于转换后的图信息向量表示,可以直接使用目前现成的机器学习算法进行训练,而不需要根据图结构信息进行重新的设计.

> PS:个人这感觉即是优点也是缺点,优点是可以有大量的现成高可用且理论相当成熟的机器学习算法,缺点是首先在进行图嵌入时可能导致部分信息的丢失,同时由于使用的模型不是专门为了图结构数据所设计的,因此可能不能很好的捕捉到图结构信息

比较容易可以看出图嵌入知识图神经网络的一个**子集**,图神经网络的主要目的是为了面对图结构数据时众多任务(task)而设计的众多神经网络形成的集合,而图嵌入技术则是能解决大部分上述图神经网络面对任务的一种方法.
这里需要注意的是对于图嵌入表示(embedding)可以使用图自编码网络生成(GAE)也可以使用非深度学习的方式进行生成(如矩阵因子分解,随机漫步)

图神经网络(GNN)与图核方法(graph kernel method)

图核方法是早期用于解决图相似度的主要技术,使用核函数衡量两张图之间的相似度,可使用监督学习的方式,也可以使用图嵌入技术通过映射函数将子图或者节点映射到向量空间.
但是与GNN有显著差异的是图核方法需要两两比对进行相似度计算,因此对于其严重受到计算性能瓶颈的影响,而GNN直接通过提取图的表示形式(representation)来进行图分类,因此更加高效.

相关符号定义:

大体上本文使用如下符号命名方式:
0.本文沿用大多数描述图的变量符号,如:V,E,G,v,e分别表示顶点集,边集,图,单个节点,单条边,同时定义$N(v)$为节点{v}的所有邻居节点构成的集合
1.大写A,X,H分别表示邻接矩阵,图特征矩阵,隐藏层特征矩阵,其中图特征包括节点特征以及边特征,因此我们使用X, X^e进行区分

IV.分类法以及相关工作介绍

按照方法分类:

- RecGNN:循环图神经网络,假定网络与邻居交换信息直到稳态平衡
- ConvGNN:卷积图神经网络,聚集自己特征x_v以及邻居特征x_u,与循环神经网络的不同在于其使用多层图卷积层提取节点的高维表示,并且其在很多复杂GNN架构中是核心组件之一.
- Graph AutoEncoder(GAE):图自编码器,使用**无监督的学习框架**,将节点或者图映射到隐含的向量表示.GAE常常用来进行图嵌入(embedding),图生成分布(?). 其通过重建图结构信息来进行向量表示,对于图生成某部分GAE可以逐渐生成(增量的形式),有的直接生成整体
- Spatial-temporal graph neural network(STGNN):时空图神经网络,其关注的重点在于同时关注时间依赖以及空间依赖关系,目前该框架的研究中通常**使用图卷积**操作用于**捕捉空间依赖**,使用RNN/CNN捕捉时间依赖.

按照层次分类:

- 节点级:主要用于节点回归,或者节点级别的分类任务,通常的做法是使用RecGNN/ConvGNN进行节点的高维表示提取,然后使用多感知机或者softmax进行端到端的转换.

- 边级:主要用于边分类,以及链接预测,使用两个节点隐藏表示作为边输入,使用相似度函数或者神经网络的方式进行预测边的标签或者边的连接强度
- 图级:输出主要用于图分类,为了获得图的向量表示,通常需要使用pooling以及readout操作

按照训练方法分类:

- 半监督的节点级分类:给定一个仅有部分节点标记的图,使用ConvGNN的方式获得对节点标签的预测模型,做法通常为使用多层卷积对有标签的节点进行训练,得到模型后对无标签节点进行提取信息后使用softmax层进行多分类.
- 有监督的图级分类:主要确定整个图的标签或者种类,主要结合图卷积层,pooling层以及readout层.图卷积层主要用于获取节点的高维表示;pooling层用于数据降维,readout层用于将图的节点表示折叠为图表示
- 无监督的图嵌入:该方法主要挖掘边级别信息,目前主要有两种方式:1.直接使用自编码框架(使用卷积层进行向量提取),将图嵌入到解码器所使用的潜在表示法.2.使用负采样的方式,定义采样的顶点对作为负采样,而图中原先存在的边作为正采样,使用逻辑回归层来区分正负采样对.

循环图神经网络(ResGNN)相关理论以及推导

循环图神经网络对于图上所有节点v**使用同样的参数**进行高维向量表示的提取,该模型的想法基于信息扩散机制,每一次使用其邻居节点的信息进行自身节点状态的更新,直到达到稳态,其更新方程表示如下:

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

参数解释:

- $f(.)$为待定的参数方程,其中为了保证收敛,$f(.)$**必须为收缩映射**,当f(.)为神经网络时,需要对雅可比矩阵施加惩罚函数(?具体方式以及原因待考证)
- $h_v^{(t)}$表示第t次迭代隐藏层参数,其中$h_v^{(0)}$随机初始化
- $x_v, x_{(v,u)}^e, x_u$,引入节点以及其邻居节点信息
- 简而言之:使用 $x_v, x_{(v,u)}^e, x_u$的信息,使用$f(.)$收缩映射对节点v进行迭代计算其隐藏向量表示.

其他操作

当收敛性得到保证之后,将得到的隐藏向量表示$h_v^{(n)}$连接到readout层进行分类.
循环图神经网络交替执行两个步骤:

- 步骤一:节点状态的传播(传给邻居节点)
- 步骤二:参数梯度计算

以此来最小化训练目标,同时使用该方法可以将GNN拓展到有环的图.(具体原理没说,我也没分析出来,暂时搁置争议).

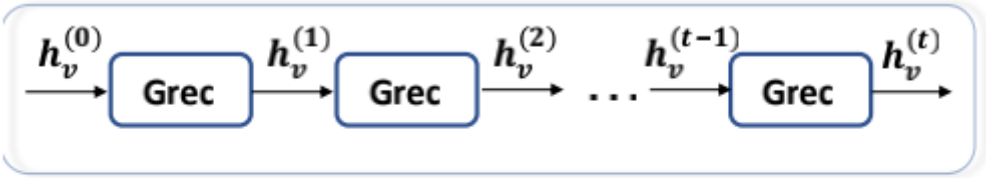
对于原始循环神经网络的优化

> - GraphESN:使用一个编码器以及一个输出层,实现了一个收缩状态转换函数,输出层训练了一个固定节点表示的作为输入(猜测为全连接层加采样),说是能提升训练效率
> - Gated Graph Neural Network(GGNN):使用一个门控循环单元(GRU)作为循环函数,减少循环次数至一个固定数量的步骤,其优势在于可以不再需要使用额外的约束参数保证收敛,其循环函数可表示为下式,剔除了x_v,$x^e_{(v,u)}$,x_u的影响,仅跟前一次迭代的自身以及邻居节点的隐向量表示有关,但是该方法无法应用到大规模图中(因为需要记录前一次迭代的所有节点隐向量信息,因此需要将所有节点信息均放置哎内存内):
>
> $$h_v^{(t)} = GRU(h_v^{(t-1)}, \sum_{u \in N(v)} W h_u^{(t-1)})$$
>
> - Stochastic Steady-state Embedding(SSE):最主要的突破在于提升了GNN对大图的拓展性,使用随机,异步的方式随机选取部分节点进行梯度计算以及更新迭代,为了算法稳定性,将历史状态与当前状态进行加权平均,其中权值α为超参数,其递推式子如下,该框架最大的缺陷为没有理论证明其收敛性:
>
> $$h_v^{(t)} = (1-\alpha)h_v^{(t-1)} + \alpha W_1 \sigma(W_2[x_v, \sum_{u \in N(v)} [h_u^{(t-1)}, x_u]])$$

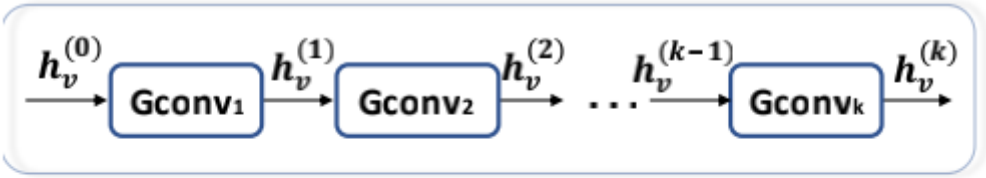
卷积图神经网络

从结构上卷机图神经网络与循环神经网络有相似之处,但是存在下面几个区别:

> - 1.通常情况下循环图神经网络**迭代次数不确**定,只有当结果趋于稳定后才终止,但图卷积神经网络**有固定的卷积层数**
> - 2.循环神经网络迭代过程中,所使用的**权重矩阵固定**,而每一层图卷积神经网络的**权重矩阵通常是不同**的,如下图对于循环神经网络均使用$Grec$一个权重函数,而卷积神经网络中$Gconv_i$均不同



(a) Recurrent Graph Neural Networks (RecGNNs). RecGNNs use the same graph recurrent layer (Grec) in updating node representations.



(b) Convolutional Graph Neural Networks (ConvGNNs). ConvGNNs use a different graph convolutional layer (Gconv) in updating node representations.

Fig. 3: RecGNNs v.s. ConvGNNs

目前对图神经网络的研究主要分为两个细分方法:

- 1.基于谱的卷积:从图信号处理的角度引入滤波器的概念,将图卷积操作解释为从图信号中移除噪音
- 2.基于空间的卷积:主要受ResGNN影响,将卷积操作定义为消息传递的过程,自[22]以来,该方法由于更好的普适性,灵活性发展很快

基于谱的卷积神经网络

由于前置知识(信号处理)没有掌握,同时由于该方法相较基于空间的卷积应用较少因此暂时跳过...

基于空间的卷积神经网络

该方法基于使用节点的空间关系(主要是边的连接关系),通过邻居节点的表示来更新中央节点的表示,该想法同样取自于ResGNN中信息传播/消息传递的想法,这里ConvGNN主要使用过边来进行节点信息的传播

基于空间的卷积神经网络相关工作:

- Neural Network for Graphs(NN4G):主要优化在于并行化了GNN,是ConvGNN第一个工作,其使用多层**参数独立**的卷积网络来学习图相互依赖,通过增量的方式来进行邻居节点的扩展,但是对于聚集操作相对简单只是将周围节点信息进行简单的加和运算.(但对于提及的resident connection以及skip connection to memorize information暂时还没有摸清,初步猜测可能是drop out操作),其迭代式子如下

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

参数解释:$f(.)$为激活函数,对于初始$h_v^{(0)}$设置为0,$W\Theta$为可学习的线性参数模型,x_v为当前节点的特征向量,但是对于第一个$\sum$循环i到k-1没有看懂做了什么操作,可能需要参考论文原文,可能是需要累加前k次迭代的结果,同时该式子还有意义相同的矩阵表示:

$$H^{(k)} = f(W^{(k)^T} X + \sum_{i=1}^{k-1} A H^{(k-1)} \Theta^{(k)^T})$$

但是该模型使用的是**非归一化的邻接矩阵**,这样可能会导致隐藏节点状态表示在尺度上有很大差距,之后Contextual Graph MArkov Model(CGMM)对其添加了概率模型,在保证空间局部性的前提下,具有更好的可解释性.

- Diffusion Convolutional Neural Network(DCNN):该模型将卷积操作是为扩散过程,将消息从一个节点向其邻居节点以概率的方式进行传播,那么进行若干论之后查看信息分布是否达到平衡,下面是其迭代式:

$$H^{(k)} = f(W^{(k)} \bigodot P^k X)$$

参数解释:$f(.)$同样是一个激活函数,W为可学习的参数矩阵,P表示消息传播的概率矩阵文中定义$P = D^{-1} A$,D为度数向量,相当于使用度数进行加权,隐藏层$H^{(k)}$与输入特征矩阵具有相同的维数
该模型最后得到$H^{(1)}, H^{(2)}, ...H^{(k)}$后将其联系(concatenation ?)一起作为输出的答案

- Diffusion Graph series(DGC):与DCNNBU不同的是,DGC最后不采用连接的方式,而是采用将每次扩散结果累加,其定义如下:

$$H = \sum_{k=0}^{K} f(P^k X W^{(k)})$$

参数解释:$W^{(k)} \in R^{D \times F}$, $f(.)$均为激活函数,使用P^k概率矩阵的幂次用来表示对于距离越远的节点所提供的概率信息越少(相当于需要借助更多的中转节点)

- Partition Graph Convolution(PGC):该算法的主要思想是将节点划分进Q组(划分方式包括但不限于最短路),然后构造Q个邻接矩阵来表示每组内已定义的邻居,然后对GCN使用不同参数矩阵对每个划分进行卷积,然后将其结果相加:

$$H^{(k)} = \sum_{j=1}^{Q} \bar{A}^{(j)} H^{(k-1)} W^{(j,k)}$$

参数解释:对于初始化$H^{(0)}=X$,$\bar{A}^{(j)}=(\tilde{D}^{(j)})^{-\frac{1}{2}}\tilde{A}^{(j)}(\tilde{D}^{(j)})^{-\frac{1}{2}}$此处D矩阵具体作用待考证,$\tilde{A}^{(j)}=A^{(j)}+I$

- PGC-DGCNN:该模型基于最短路,在DGC的基础上调整了远邻居节点的贡献,其框架定义了最短路矩阵$S^{(j)}$,当节点v到节点u为最短路为j时,$S_{v,u}^{(j)}=1$,通过超参数r控制接收域大小,其卷积操作定义为:

$$H^{(k)}=||_{j=0}^{r}f((\tilde{D}^{(j)})^{-1}S^{(j)}H^{(k-1)}W^{(j,k)})$$

参数解释:$\tilde{D}_{ii}^{(j)}=\sum_l S_{i,l}^{(j)}$大致上意思应该是计算第j次迭代时节点i能到达的节点数,类似于之前度数矩阵的作用,对于初始化为$H^{(0)}=X$,而操作$||$表示向量的连接,不过需要注意的是由于需要计算任意点之间的最短路,因此需要额外计算上$O(n^3)$的复杂度

- **Message Passing Neural Network(MPNN)**:这篇文章的主要贡献在于使用形式化语言对基于空间卷积的图神经网络提出了一个普遍的框架,首先假定图卷积操作为通过边的信息传递,那么消息传递方程(或者说图卷积函数)为:

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

对于$h_0=x_v$,对于函数$U_k(.),M_k(.)$均为可学习的参数矩阵,当得到节点的隐藏表示$h_v^{(K)}$之后可以将其输送到输出层以完成节点级别的预测任务,也可以将其输入到readout层进行图级别的预测,对于readout函数生成整张图的表示同样也可以用非形式化语言定义为:

$$h_G=R(h_v^{(K)}|v\in G)$$

其中$R(.)$为带有可学习参数的readout函数
但是Graph Isomorphism Network(GIN)文章中提出上述模型不适合处理区分不同图结构产生的图嵌入这种问题,因此他们使用可学习的参数$\epsilon^{(k)}$调整中心节点的权重进行优化,其优化后的式子如下:

$$h_v^{(k)}=MLP((1+\epsilon^{(k)})h_v^{(k-1)}+\sum_{u\in N(v)}h_u^{(k-1)})$$

其中$MLP(.)$设定为多层感知机

- GraphSage:由于节点的邻居节点数量不固定,且通常跨若干数量级,因此每次计算单个节点的所有邻居通常情况下不高效,因此该篇文章引入采样的思想,每次选择固定数量邻居借点进行结算,

其图卷积函数如下:

$$h_v^k = \sigma(W^k \cdot f_k(h_v^{k-1}, \{\forall u \in S_{N(v)}\}))$$

参数解释:初始化$h_v^{(0)} = x_v$,设$f_k(.)$为聚集函数,$S_{N(v)}$为随机采样节点v的邻居节点,这里需要保证聚集函数是与被操作数的排列顺序无关的函数(由于使用随机采样)

- Graph Attention Network(GAT):图注意力机制主要学习节点间相对权重关系,其图卷积操作定义为:

$$h_v^k = \sigma(\sum_{u \in N(v) \cup v} \alpha_{vu}^{(k)} W^{(k)} h_u^{(k-1)})$$

其中定义$h_v^{(0)} = x_v$,注意力权重$\alpha_{vu}^{(k)}$用来衡量节点v与邻居节点u的连接强度其表达式可以写为:

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

参数解释:$g(.)$是LeakyReLU激活函数,a为一个可学习的向量参数,softmax函数保证所有邻居节点注意力权值和为1.
在该模型的基础上可以进一步推广为多注意力头的模型,进一步提高模型的表达能力,该模型对于节点级别的分类任务较GraphSage提升很大.
在该模型的基础上Gated Attention Network(GAAN)针对原先多注意头模型中每个注意力头地位相同的现象,为每个注意力头额外添加了一个权值来进行平衡,也有人进一步通过一个类LSTM闸门机制控制消息的流动.

- Mixture Model Network(MoNet):该模型引入一个伪坐标的概念,用于确定节点之间的相对位置关系,当相对位置确定后,再通过映射函数确定相对权值,这样做的好处是可以将图过滤器的参数共享给整张图,该模型同时还提出了一个高斯内核的方法使用可学习的参数来自适应的学习权重函数.
- PATCHY-SAN:该模型使用某些标准加可学习的权重来对节点的邻居节点进行排序,并通过排序将权重进行分享.通过选择前k个邻居使得节点仅有固定个数的有序邻居节点,节点可以将图结构转化为网格化结构,使得可以使用标准的一维卷积过滤器将邻居特征进行聚集,但该种方法对topk的处理需要大量计算以及数据预处理,因此有人提出 Graph Convolutional Network (LGCN)通过对邻居节点的特征信息进行排序(不是对节点进行排序),如将特征矩阵每列分别排序后前q行特征作为输入数据

提高训练效率的方法:

通常情况下GCN需要对整张图以及训练的中间状态进行存储,将其存放在内存中,因此**内存溢出问题对于GCN来说非常棘手**,

> - GraphSage:提出里一种批处理训练算法:通过采样的方式**递归的扩展根节点**的邻居节点,通过k轮采集固定大小的子图,使用其进行层次聚合生成该根节点的隐藏表示.
> - Fast Learning with Graph Convolutional Network (Fast GCN)通过采样固定个数节点(不要求由根节点生成),将图卷积解释为节点的**图嵌入表示在概率意义下的积分变换**,同时使用近似以及方差减小技术促进训练过程,但由于随机采样因此对于相邻网络层之间存在潜在的稀疏表示,因此有人使用自适应的分层采样的方式,下层节点通过上层节点进行条件选择,增强了节点之间的联系因此提高了准确性.
> - Cluster-GCN:使用图聚类算法进行采样,在采样的子图中进行图卷积操作,由于操作限制在被采样的子图中,因此该方法可以处理更大的图或者更深的网络结构

图池化模块:

直接使用全部生成的节点特征,对于计算来说是具有很大挑战的,因此需要使用数据降维的方式来减少计算量,这样的数据降维操作有两个名称不同但原理一直的应用

> - 图池化操作(pooling operator):该操作主要针对生成节点更小的表示来进行数据降维,减少参数个数,以此来避免过拟合,排列不变形以及计算复杂度过高
> - 图读出操作(readout):该操作主要基于节点级表示,生成图级别的表示

上述操作以下我们均使用池化(pooling)代替,对于早期研究,图粗化算法使用**基于其拓扑结构**的特征分解到凝聚图(coarsen graph),但是该方法对计算的复杂性要求很高,因此使用格拉克鲁斯算法(Graclus algorithm)作为计算原始图的密集表示的一种替代方法:

> Graclus algorithm:基于Metis，是用一个贪心算法来计算给定图的粗化结果，能够最小化一些流行的谱聚类的结果

目前使用的数据降维的最原始也最有效的方法包括平均,最大值,总和(mean/max/sum)并且这些方法处理速度更快.
有人通过研究发现在网络开始使用一个简单的max/mean池化在减少数据维度以及减轻图傅里叶变换过程中高昂的开销起到了积极的作用.也有一些工作使用图注意力机制来提高mean/sum池化效果.
但即使使用注意力机制,上述机制对于生成一个与图大小无关的固定大小embedding低效,因此不能满足需求.

> - Set2Set:当输入大小增长时生成一个随之增长的内存,用于实现一个LSTM来将即将销毁的依赖于顺序的信息以embbeding的形式集成到内存中
> - ChebNet:通过一种有意义的方式重新安排图中节点来进行高效池化,首先使用Graclus算法将输入图粗化为多个级别,粗化之后**使用平衡二叉树**将相似的节点安排在一起,这样的重新安排的信号较初始池化效率更高.

- DGCNN:提出了一种SortPooling策略,将节点重新排列为一个有意义的顺序来进行池化,不过是根据图的结构作用,将无序节点特征从空间图中得到的卷积视为连续的WL颜色,并用其作为节点排序,除此之外还通过截断/拓展节点特征矩阵将图大小统一为q
- DiffPool:为了解决仅注重图特征进行池化,而忽视了图结构信息,该方法生成图的不同层次(layer)表示,不直接聚集几点,而是通过学习的方式得到一个团的赋值矩阵,其中一个团表示一层的所有节点,该矩阵表示用过下面的方式进行生成:

  $$S^{(k)} = softmax(ConvGNN_k(A^{(k)}, H^{(k)}))$$

  采用这种方法目的在于学习复杂的节点拓扑结构以及特征信息,但是因此会生成稠密图,且计算复杂性会达到$O(n^2)$
- SAGPool:最近最新提出的方法,使用自注意力机制学习节点特征以及图拓扑结构

GNN理论基础

接受域(receptive field): 对于节点来说接受域为一些节点的集合,用于表示该节点的最终表示,使用多层空间涂卷机层,那么每次接受域都会往+1的邻居进行拓展,有人证明对于任意节点,其接受域可以拓展到包含所有节点,因此该性质允许堆叠卷积层数目来来获取全局信息
VC维(Vapnik-Chervonenkis Dimension): 用于衡量模型的分类能力,能打散的最大节点数目(相当于能分类的最大种类),对于给定参数个数为p,节点个数为n的GNN,如果使用sigmoid或者切向双曲激活函数则其分类能力可以到达$O(p^4n^2)$,如果使用分段多项式激活函数,其分类能力仅能达到$O(p^2n)$
图同构(Graph isomorphism): 使用GNN将两张图分别进行图嵌入,那么可以通过Weisfeler-Lehman测试来判断两张图是否同构,如果聚合函数以及独处函数是内射的(injective)的那么GNN也可以像WL一样高效区分两张图
等边性以及不变性(Equivariance and invariance): 对于节点级任务GNN必须是一个等变函数(equivariant function),对于图级别任务必须是一个不变性函数(invariant function).对于等变性与不变性的定义如下:

- 等变性(equivariant):假设$f(A, X) \in R^{n\times d}$是一个GNN,Q为节点顺序的任意一个排列,满足下面这样式子则称该GNN满足等变性

  $$f(QAQ^T, QX) = Qf(A, X)$$

- 不变性(invariant):满足下面式子

  $$f(QAQ^T, QX) = f(A, X)$$

通常情况下为了满足等变/不变性,因此通常情况下会选择与**节点顺序无关**的方式来组成GNN
通用近似能力: RecGNN可以以任意精度近似任何函数,ConvGNN在信息传递模型下的近似能力无法近似定义在多重集合上的连续函数,但是对于满足不变性(invariant)网络中ConvGNN可以逼近任何定义在图上的任意满足不变性的函数

图自编码器

图自编码器使用深度学习架构,将节点映射到一个潜在向量空间,并提供一个从这样一个潜在向量空间中解码图信息的方式.可以用于学习网络嵌入模式,也可以用于生成新的图.

网络嵌入(Network Embedding)

网络嵌入是将节点的拓扑关系映射到一个低维向量,GAE使用一个编码器用于提取网络嵌入,并使用一个解码器用于还原(?)图拓扑关系(如邻接矩阵或者PPMI矩阵)

- Deep Neural Network for Graph Representations(DNGR):使用堆叠的去噪自编码器对PPMI矩阵通过多层感知机进行编码以及解码
- Structural Deep Network Embedding (SDNE):使用堆叠的自编码器保留节点的第一维以及第二维接近,该模型对输出编码器以及解码器使用两套不同的损失函数,对于编码器其损失函数通过减小节点网络嵌入与其邻居节点嵌入的距离,保存节点第一维,其第一个损失函数定义如下:

$$L_{1st} = \sum_{(u,v)\in E} A_{u,v}||enc(x_v) - enc(x_u)||^2$$

相关参数定义如下:$x_v = A_v$,$enc(\cdot)$定义为多层感知机组成的编码器
对于第二个损失函数定义为:

$$L_{2nd} = \sum_{v\in V} ||(dec(enc(x_v)) - x_v) \odot b_v||^2$$

其中参数本文并不是特别清晰(?),需要看一下原文章才能确定参数具体含义,但可以确定的是dec是一共用于解码的多层感知机

- Graph Autoencoder(GAE):除了采集图拓扑信息,还通过图卷积搜集节点信息用于嵌入,其使用两个图卷积层用于搜集节点信息:

$$Z = enc(X, A) = Gconv(f(Gconv(A, X; \Theta_1)); \Theta_2)$$

其中Z代表图前途矩阵,$f(\cdot)$表示ReLU激活函数以及$Gconv$表示图卷积层函数(其使用基于谱的卷积函数),其解码器使用邻接矩阵还原节点连接关系:

$$\hat{A}_{v,u} = dec(z_v, z_u) = \sigma(z_v^T z_u)$$

其中z_v表示节点v的嵌入表示,GAE训练过程中训练目的为减小原始图邻接矩阵与重建图邻接矩阵的负交叉熵.

- Variational Graph Autoencoder (VGAE):由于自编码器容量过大,因此单纯使用邻接矩阵用于重建图会造成过拟合现象,因此该模型对GAE使用变分的方式来学习数据的分布式表示,其优化变分下界L为:

$$L = E_{q(Z|X,A)}[log\, p(A|Z)] - KL[q(Z|X,A)||p(Z)]$$

其中$KL(\cdot)$为相对熵用于描述两分布之间距离,$p(Z)$表示高斯先验,公式比较复杂,有需要的话可以查看原文

- Adversarially Regularized Variational Graph Autoencoder(ARVGA):使用生成对抗网络(GAN)的方式,在生成器以及辨别器,生成器用于生成假样本,分辨器其用于将其分辨出来,ARVGA则通过学习一个编码器,使得其产生的经验分布与先验分布难以区分(相当于生成器)
- GraphSage:使用两层卷积,不过其使用下面的损失函数保留两个节点之间的关系信息,而不是优化重建误差:

$$L(z_v) = -log(dec(z_u, z_v)) - QE_{v_n\ P_n(v)} log(-dec(z_v, z_{v_n}))$$

其中节点u是v的邻居节点,v_n是v的一个遥远节点,并是由负采样分布$P_n(v)$生成的,Q表示负样本数量,该损失函数强制要求相邻节点由相似的节点表示,遥远节点有不同的节点表示,DGI使用局部网络嵌入的方式通过最大化局部共有信息来抓住全局结构信息,在实验水平上对GraphSage有明显的效果提升.

- Deep Recursive Network Embedding (DRNE):由于图样本的数目较少,因此正采样与负采样数目不成比例,因此为了缓解数据稀疏性,主要采用将图转化为随机排列或者随机漫步的方式进行采样,这样针对序列的学习方式就可以直接使用,因此该模型使用长短期网络来聚合节点信息,其重构误差定义为:

$$L = \sum_{v \in V} ||z_v - LSTM(z_u|u \in N(v))||^2$$

其中z_v表示通过字典查找的节点v的网络嵌入模式,LSTM网络中使用节点v邻居按节点度数的随机排列,因此该模型隐式的使用LSTM学习网络嵌入模式,而非生成网络嵌入,因此避免上述要求对排列不变性的要求

- Network Representations with Adversarially Regularized Autoencoders (NetRA):该模型使用了通用损失方程,提出了一个编码/解码框架:

$$L = -E_z\ P_{data}(z)(dist(z, dec(enc(z))))$$

其中$dist(\cdot)$ 表示节点嵌入表示与重建后的表示之间的距离,其编码器/解码器使用LSTM,以节点v作为起点的随机漫步作为输入,与ARVGA类似使用对抗训练的方式获得先验分布.

图生成(Graph Generation)

对于大量分布近似的图结构,GAE可以学习这些图的分布特征,目前GAE用于生成图的主要目的在于解决分子图生成(molecular graph generation?),其在药物发现中有很高的实用价值,GAE可以以按步骤或者全局的方式生成一个新的图

- SMILES:使用CNN以及RNN分别作为编码器以及解码器独立的生成分子图的字符串表示,但是其只针对特定的专用领域,不具有良好的推广能力
- Deep Generative Model of Graphs (DeepGMG):对于一般意义上的图通过迭代添加节点以及边的方式来生成满足一定条件的图,其首先假设一个图的概率是所有可能的节点排列的集合:

$$p(G) = \sum_{\pi} p(G, \pi)$$

  其中π表示一个节点的排列,其获取所有节点以及边的复合概率,其通过做出一系列决策来生成图:1.是否添加一个节点?2.添加哪个节点?3.是否添加一条边?4.将哪个节点与新的节点连边?通过当前节点状况以及当前图状况有条件的通过ResGNN进行更新
- GraphRNN:使用一个图级别RNN,以及一个边级别的RNN来建模节点以及边的生成过程,图级别的GNN每次向节点序列中添加一个新的节点,边级别RNN生成一个二进制串用于表示新节点与之前节点的连接强度.
- Graph Variational Autoencoder (GraphVAE):该模型将边以及节点作为独立的随机变量进行处理,假设使用编码器生成的后验分布为$q_\phi(z|G)$,使用解码器生成的分布为$q_\phi(G|z)$,该模型的优化目标为:

$$L(\phi, \theta; G) = E_{q_\phi(z|G)}[-log\, p_\theta(G|z)] + KL[q_\phi(z|G)||p(z)]$$

  其中$p(z)$表示高斯分布,ϕ, θ表示科学系变量,使用convGNN作为编码器,以及多层感知机作为解码器,该模型将生成图的邻接矩阵,节点属性,边属性作为输出,但是对于生成图的全局属性并不能很好处理(如连通性,有效性节点兼容性)
- Regularized Graph Variational Au toencoder (RGVAE):进一步使用图变分自编码事假有效性约束来曾泽华解码器的输出分布
- Molecular Generative Adversarial Network (MolGAN):使用强化学习的方式生成满足需求的目标生成,使用一个生成器以及辨别器的方式来进行竞争训练,生成器生成假图,辨别器使用历史数据进行辨别,并且对辨别器引入奖励网络,来鼓励生成满足特定性质的某些图
- NetGAN:使用LSTM以及 Wasserstein GAN来使用基于随机游走的方式生成图,使用LSTM训练一个生成器用于生成合理的随机游走序列,使用辨别器来识别这些生成的随机游走,最后使用由生成的随机游走序列得到正则化的共现矩阵(?),但是这种方式会因为环的出现损失一定结构信息,并且没有很好的拓展到大图的能(因为输出空间可能很大)

基于时空的图神经网络(STGNN)

由于在真实世界应用过程中,图结构通常是动态变化的,因此时空图神经网络主要注重抓住图的动态变化,主要在节点之间相互依赖关系确定的情况下对动态节点输入进行建模.
STGNN同时抓住一张图的时间依赖关系以及空间依赖关系,其计算任务可以用来预测未来节点值或者标签的变化,或是预测时空图的标签,STGNN使用两种基础研究方向:基于RNN的,以及基于CNN的.

- 基于RNN的方法:铜鼓哦使用输入过滤以及隐藏状态抓住时空依赖关系没并将其传到一个循环的图卷积网络.基本上可以使用下面的范式来进行解决:

$$H^{(t)} = \sigma(WX^{(t)}) + UH^{(t-1)} + b)$$

  其中$X^{(t)} \in R^{n \times d}$表示在t时刻节点特征矩阵,插入的图卷积可以表示为:

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

  其中$Gconv(\cdot)$表示图卷积层,如GCRN结合LSTM以及ChebNet.DCRNN提出了将扩散图卷积层合并到GRU网络中,使用一个编码/解码框架来预测未来K步节点值的变化.
  另一个平行工作是使用节点级别的RNN以及边级别的RNN来掌握不同层次的时空信息.如Structural-RNN:提出了一个循环框架用于预测每一次节点标签的变化,其包含2种RNN,一种是节点级别的RNN,另一种为边级别的RNN,对于节点/边信息分别传入各自框架,为了合并各自得到的信息,节点RNN的输出将作为edge输出的一部分,但由于对节点/边使用了不同的RNN,因此显著提升了模型的复杂性,这里该模型将其划分为语义组,对于划分为相同语义组的节点以及边使用相同的RNN模型,以此来减少计算开销.
  但是基于RNN的方法有相同的缺陷就是需要大量的时间进行迭代传播,并且存在梯度爆炸/消失的问题.
- 基于CNN的方法:CNN就是用于解决RNN方法的计算复杂性等问题的一种替代,由于使用非递归的方式进行迭代,因此具有并行计算的基础,情切有稳定梯度以及较少的内存开销要求.
  基于CNN的方式将一维CNN层以及图卷积层进行相交,使得可以各自学习时间以及空间依赖关系$\chi \in R^{T \times n \times d}$,一维CNN划过$\chi_{[:,i,:]}$用于聚集时间信息
- CGCN:使用一维卷积层以及ChebNet或者GCN层,按顺序通过堆叠一个一维门控层,一个图卷积层,另一个门控卷积层来构建时空块
- ST-GCN:用一维卷积层以及一个PGC层构建一个时空块
- WaveNet:提出了一个自适应的邻接矩阵用于实现图卷积,目的在于通过众多快照的方式来学习隐含的静态图表示,而不需要给定图的邻接信息,其自适应邻接矩阵定义如下:

$$Aadp = SoftMax(ReLU(E_1 E_2^T))$$

  其中SoftMax函数用于计算行向量,E_1表示源节点的嵌入形式,E_2表示目标节点的嵌入形式,通

> 过两项的成绩可以获得源节点以及目标节点之间的依赖权重,通过一个复杂的基于CNN的时空图神经网络,该模型可以在不给出邻接矩阵的情况下得到不错的结果
> - GaAN:使用注意力机制通过基于RNN的方式来学习动态空间依赖关系,使用注意力函数使用当前两端节点作为输入来更新边权重
> - ASTGCN:进一步使用空间注意力机制,以及时间注意力机制使用基于CNN的方式来学习隐含的动态空间/时间依赖,但是学习隐含空间依赖的共同缺点就是需要计算节点对的空间依赖权重,这个代价为$O(n^2)$

V.递归学习:

图读出操作(Readout)

别名:图粗化/图池化
目的:得到每个节点的表示之后,生成整个图的表示
要求:操作需要对**操作顺序无关**,因为存在图同构现象(Graph Isomorphism)

Readout操作目前主要分为基于统计的方法,以及基于学习的方法,下面简单介绍一下两种方法:

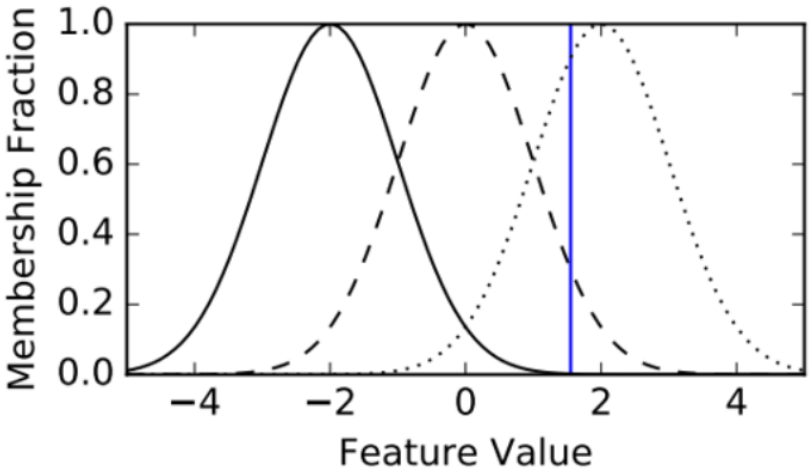
基于统计的方法(Statistics Category)

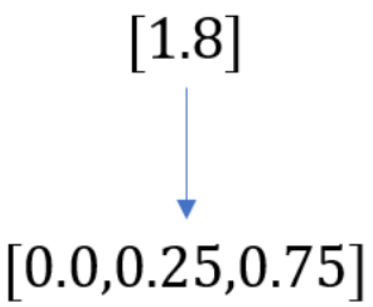
基于统计的方法实现较为简单有效,但是存在下面问题

> - 1.使用基于统计的方法往往会导致信息损失过大,原因是原本庞大的数据空间例如100个节点每个节点100维,通过简单的统计函数直接将信息压缩到了很小的空间
> - 2.在数据压缩的过程中会导致某一维上数据分布特性被完全抹除
> - 3.对于所有节点一视同仁,对于某些重要结点信息更多更重要的无法做到更多的表示.

优化方式:在数据信息压缩和增强之间进行平衡,可以使用直方图分段的方式,分段统计信息,在数据压缩的前提下,使用额外的空间保留部分分布信息,或者使用高斯函数实现模糊直方图(Fuzzy Histogram)

递归第二层
高斯函数和模糊直方图:定义特征值区域边界点为高斯根部的均值,并设置好方差,然后作x=特征值平行y轴的直线与设定的高斯分布求交点,将得到的交点纵坐标向量化即得到了给定特征值的模糊直方图(原理不明),下面是一个特征值为1.8到模糊直方图向量[0.0,0.25,0.75]的映射过程





基于学习的方式

基于学习的方式寄希望于神经网络来拟合这个过程,使得其参数化,主要由下面的几种方法:

采样加全连接(Sample And FC): 取固定数目的节点,通过一个全连接层得到图表示可以使用随机采样,规则采样得到确定数目的节点,使用下面的公式来进行计算

$$h_G = FC(H^L)$$

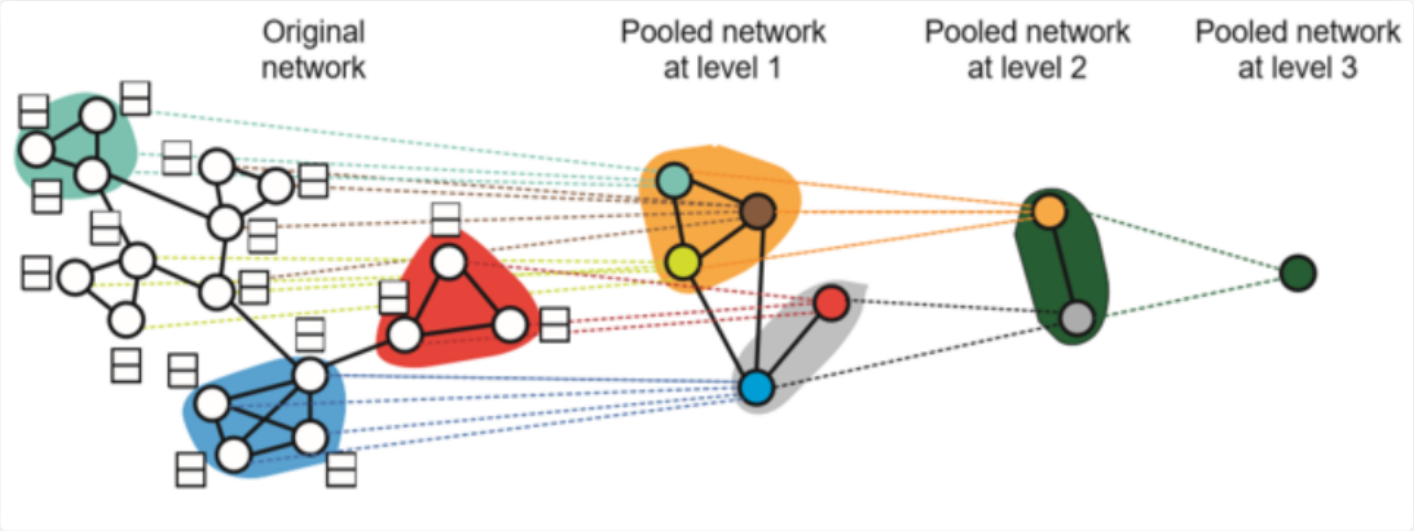
其中H^L表示将采样的节点进行拼接,但是该方法**很难适用于规模差距很大的图**,即图节点个数跨数量级的现象较为显著时,无法设定一个合理的采样方式以及大小.

全局节点(Global Node):

引入一个全局节点作为这张图的根节点,将其与图中每个节点通过特殊的边进行连接,最终拿该节点作为整张图的表示.(具体方式查到的资料没有细说)

可微池化(Differentiable Pooling)

上述方法均较为简单,无法层次化的获取图表示,"Graph Representation Learning with Differentiable Pooling"该文中提出通过一个逐渐压缩信息的方式来获取图的最终表示,如下图:



其使用了两个不共享参数的GCN模块SC,NR分别完成结点聚类(Soft Clustering)与结点表示(Node Representation)

附录

符号表:

符号	含义
$\|.\|$	集合大小
$\odot$	逐元素相乘
G	图
V	节点集合
v	单个节点$v \in V$
E	边集合
$e_{i,j}$	一条边$e_{i,j} \in E$
$N(v)$	v的邻居节点集合
A	图的邻接矩阵
A^T	图邻接矩阵的转置
A^n	邻接矩阵A的n次方

符号	含义
$[A, B]$	连接矩阵A,B
D	A的度数矩阵$D_{i,i} = \sum_{j=1}^{n} A_{ij}$
n	节点个数$n = \|\|V\|\|$
m	边个数$m = \|\|E\|\|$
d	点的特征向量维数
b	隐藏层节点特征向量的维数
c	边的特征向量维数
$X \in R^{n \times d}$	图特征向量矩阵
$x \in R^{n}$	图特征向量矩阵的某一列
$x_v \in R^{d}$	某节点的特征向量
$X^{e} \in R^{m \times c}$	图边特征矩阵
$X^{e}_{(v,u)} \in R^{c}$	边(v, u)的特征向量
$X^{(t)} \in R^{n \times d}$	在第t次迭代时节点的特征矩阵
$H \in R^{n \times b}$	隐藏层节点的特征矩阵
$h_v \in R^{b}$	隐藏层节点v的特征向量
k	层的标号,表示第k层网络
t	迭代的标号,表示第t次迭代
$\sigma(.)$	sigmoid激活函数
$\sigma_h(.)$	切线双曲激活函数
W, Θ, w, θ	可学习的线性模型参数