GNN <https://mp.weixin.qq.com/s/sg9O761F0KHAmCPOfMW_kQ>

GNN的本质就是提取图拓扑空间特征，，目前有两种主流方式来提取图空间特征

<https://blog.csdn.net/yyl424525/article/details/103172141>

1. **spatial domain空间域**，是非常直观的一种方式

顾名思义：提取拓扑图上的空间特征，那么就把每个顶点相邻的neighbors找出来

1. 按照什么条件去找中心**spatial**的neighbors，也就是如何确定感受域
2. 按照什么方式处理包含不同数目neighbors的特征
3. **spectral domain频谱域** 就是GCN的理论基础了这种思路就是希望借助图谱的理论来实现拓扑图上的卷积操作。

频谱型：基于频谱的方法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 算被解释为从图信号中去除噪声。

但是这种基于谱域的GCN有很大的局限性，比如不能处理有向图，扩展性差，不能处理大图等，目前基于空间的方法因其具有良好的效率、灵活性和通用性，近年来发展迅速

GNN主要类型

1. 卷积。Graph Convolutional Network（GCN）希望将卷积操作应用在图结构数据上，主要分为Spectral Method和Spatial Method（Non-spectral Method）两类。Spectral Method希望使用谱分解的方法，应用图的拉普拉斯矩阵分解进行节点的信息收集。Spatial Method直接使用图的拓扑结构，根据图的邻居信息进行信息收集。
2. 注意力机制。Graph Attention Network 致力于将注意力机制应用在图中的信息收集阶段。
3. 门机制。这些变体将门机制应用于节点更新阶段。Gated graph neural network 将 GRU 机制应用于节点更新。很多工作致力于将 LSTM 应用于不同类型的图上，根据具体情境的不同，可以分为 Tree LSTM、Graph LSTM 和 Sentence LSTM 等。
4. 残差连接。注意到堆叠多层图神经网络可能引起信息平滑的问题，很多工作将残差机制应用于图神经网络中，文中介绍了 Highway GNN 和 Jump Knowledge Network 两种不同的处理方式。

## 基于频谱域的GCN的基本原理，基本使用

来自知乎的解释：

<https://zhuanlan.zhihu.com/p/58178060?utm_source=wechat_session&utm_medium=social&utm_oi=685930719541465088>

https://zhuanlan.zhihu.com/p/71200936?utm\_source=wechat\_session&utm\_medium=social&utm\_oi=685930719541465088

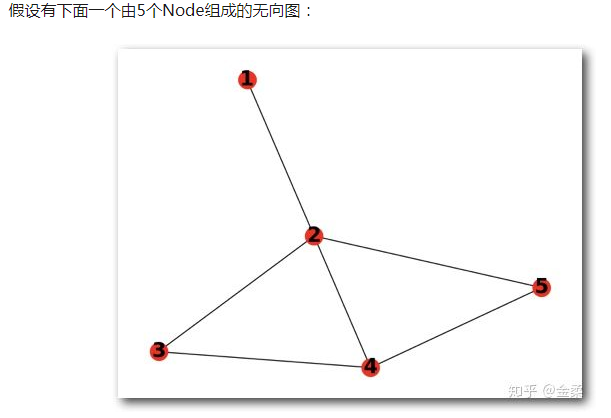
GCN下的embedding的思想是，每一个节点聚合其领域节点以及自己的特征数据作为当前节点的特征，怎么学习的：

邻接矩阵：A， 特征数据: X 单位矩阵：I 度矩阵：D

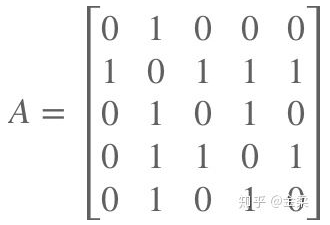
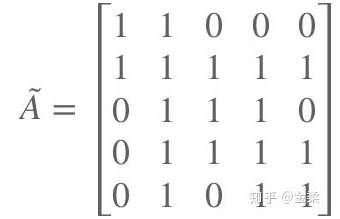
从数学上看 A\*X就可以聚合当前节点的一阶领域节点的特征和但是没有自身特征参与，

(A+I)\*X可以聚合自身节点以及邻居节点的特征，

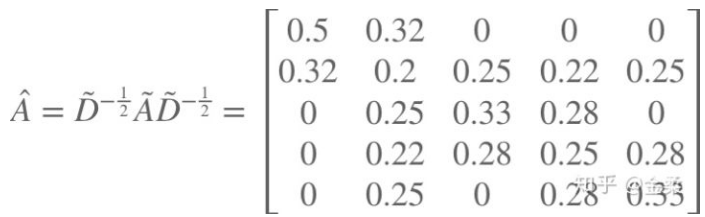
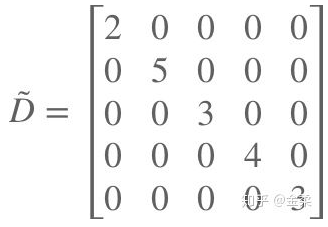
D\*\*-1\*A\*X 可以将特征归一化，这一步参考拉普拉斯矩阵分解



邻接矩阵A 邻接矩阵A的基础上加上自环（单位对角阵I）

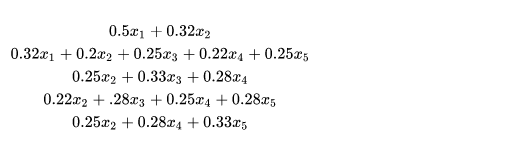
度矩阵 卷积操作



节点特征（一维）



在神经网络第一层的传播中，输入是节点的特征。为了方便，假设每个Node的特征向量是1维的，于是X=(x1,x2,x3,x4,x5)'，将特征矩阵左乘 [公式] ，即可以得到变换后的特征：



可以看到，对比一个普通的NN，GCN只是在特征上做了一个变换，而这个变换的实质就是特征通过拓扑结构进行了传播。每个节点的特征不再是自身的特征，而是自身和其邻居节点的特征加权求和。讲到这里，大家就很清楚为什么加自环的吧，其目的是需要在特征变换时保留了自身特征的贡献。

D\_hat\*\*-1 \* A\_hat \* X 已经完成了一阶领域的特征聚合，这就是图神经网络中的卷积操作

对于卷积中的感受域来说这里的感受域就是一阶邻居，全局就是整个图，经过二次卷积，感受域扩大一倍。

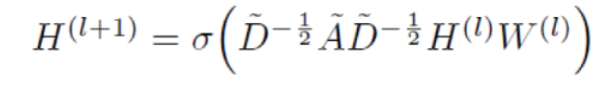
一个隐藏层的完整表达就是：relu(D\_hat\*\*-1 \* A\_hat \* X \* W)

D\_hat 具有自环的度矩阵， A\_hat 自环的邻接矩阵 W：权重系数， relu(): 激活函数。

在神经网络第二层的传播中，节点依然会按照上述机制进行特征传播，但这里的特征不再是节点的原始特征，而是经过变换后（传播->聚合->Emdedding->非线性变换）的特征。这个时候，每个节点可以接收到2-hop的信息，感受域进一步增加。

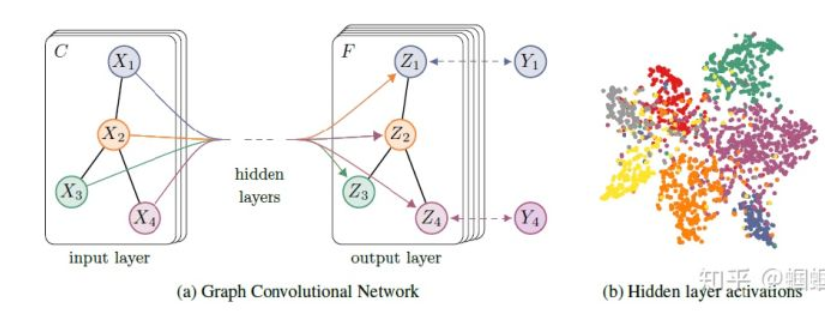
假设我们手头有一批图数据，其中有N个节点（node），每个节点都有自己的特征，我们设这些节点的特征组成一个N×D维的矩阵X，然后各个节点之间的关系也会形成一个N×N维的矩阵A，也称为邻接矩阵（adjacency matrix）。X和A便是我们模型的输入。

它的层与层之间的传播方式是

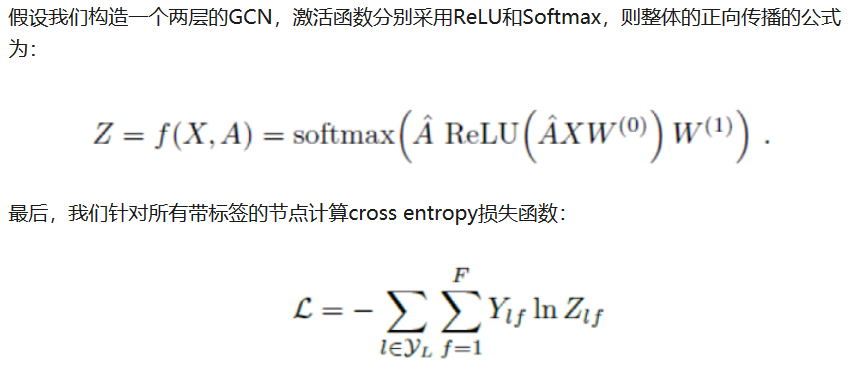


我们先不用考虑为什么要这样去设计一个公式。我们现在只用知道： 这个部分，**是可以事先算好的**，因为D波浪由A计算而来，而A是我们的输入之一。

在整个训练过程中应该是固定的共享的，他定义了节点特征的聚合范式

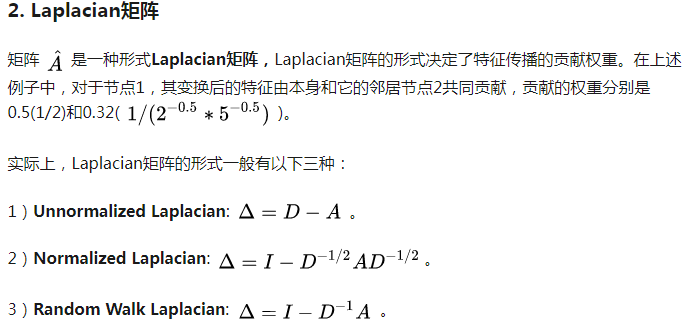


上图中的GCN输入一个图，通过若干层GCN每个node的特征从X变成了Z，但是，无论中间有多少层，**node之间的连接关系，即A，都是共享的**。



就可以训练一个node classification的模型了。由于即使只有很少的node有标签也能训练，作者称他们的方法为**半监督分类**。

当然，你也可以用这个方法去做graph classification、link prediction，只是把损失函数给变化一下即可。



邻接矩阵A的基础上加上自环（单位对角阵I）现在的问题是为什么在GCN中要使用Normalized Laplacian，这里我们只从直觉上说明这个问题。

假设节点A的邻居只有B，为了对A进行特征传播，一个最直接的聚合方法是就是平均贡献，即：X=0.5A+0.5B（X=1/degree(A)\*A+1/degree(A)\*B）。这样的做法看起来很有道理，但是如果B的邻居非常非常多（极端情况是与图上其他所有点都连接），那么经过特征变换后图上很多点的特征都会非常像相似，因此显然在传播的时候需要考虑邻居的度。

因此，我们可以将传播公式改写成X=1/degree(A)\*A+1/degree(B)\*B，然而这样的问题在于若degree(B)很大，B的贡献将几乎忽略不计。综合上述两种形式，既不会让邻居B的贡献过大、也不会让邻居B的贡献过小，于是得到传播公式：

X=1/degree(A)\*A+1/(sqrt(degree(A)\*degree(B))\*B

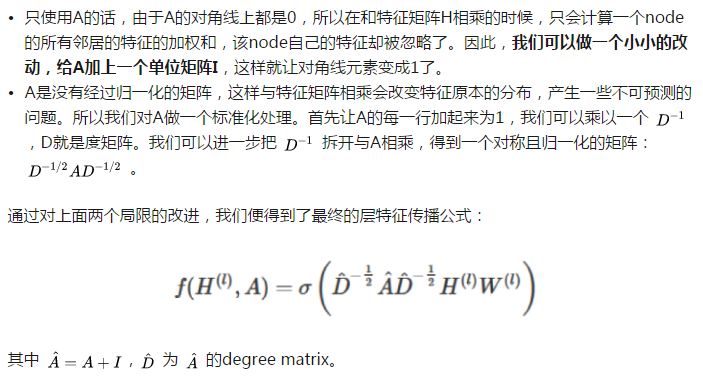
这种传播公式对应的变换矩阵就是Normalized Laplacian。

如果还不懂的话，我们来从头假设一下，如果我要去将神经网络应用与图数据上面会怎么做。

Ok 我们每一层的输入依然是邻接矩阵A和节点的特征H，那么我们直接做一个内积，然后再乘以一个参数数据W 在激活一下的话，一个隐藏层就完成了，



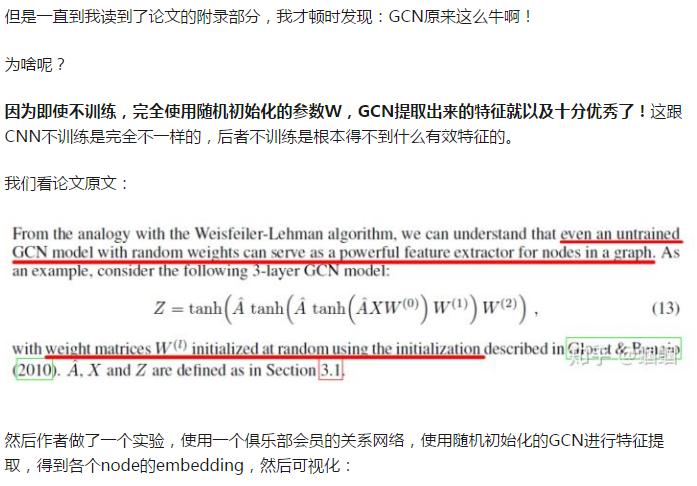
这样也可以完成领域信息的聚合，不足之处在于

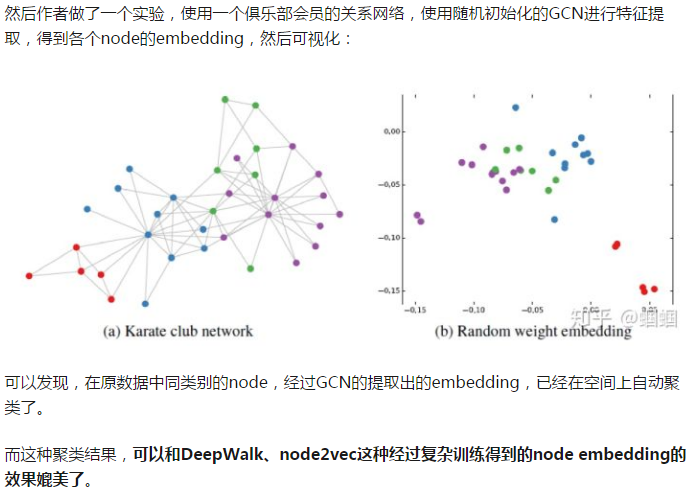


GCN为何叫GCN，其缘由就在于此，就是使用对称归一化拉普拉斯矩阵。

### GCN有多牛

在看了上面的公式以及训练方法之后，我并没有觉得GCN有多么特别，无非就是一个设计巧妙的公式嘛，也许我不用这么复杂的公式，多加一点训练数据或者把模型做深，也可能达到媲美的效果呢。





说的夸张一点，比赛还没开始，GCN就已经在终点了。看到这里我不禁猛拍大腿打呼：“NB！”

还没训练就已经效果这么好，那给少量的标注信息，GCN的效果就会更加出色。

作者接着给每一类的node，提供仅仅一个标注样本，然后去训练，得到的可视化效果如下

其他关于GCN的点滴：

1. 对于很多网络，我们可能没有节点的特征，这个时候可以使用GCN吗？答案是可以的，如论文中作者对那个俱乐部网络，采用的方法就是用单位矩阵 I 替换特征矩阵 X。

2. 我没有任何的节点类别的标注，或者什么其他的标注信息，可以使用GCN吗？当然，就如前面讲的，不训练的GCN，也可以用来提取graph embedding，而且效果还不错。

3. GCN网络的层数多少比较好？论文的作者做过GCN网络深度的对比研究，在他们的实验中发现，GCN层数不宜多，2-3层的效果就很好了

4. GCN的消息传播思想和pagerank思想也有相似之处，两者的本质都是通过边的权值计算节点的重要程度，并通过迭代（GCN 里是多层卷积）累积重要程度直至收敛。所以说，在社交网络中同样可以使用GCN来跑传播算法，将黑节点特征置为2，进行半监督学习。

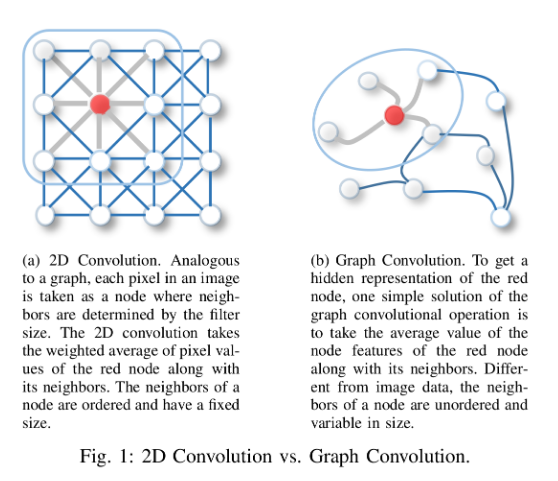
<https://mp.weixin.qq.com/s/MOPaKw4HqHDlnDqVUzdVvA>

1. 通常GCN这样的训练函数需要将整个图数据和所有节点中间状态保存到内存中。特别是当一个图包含数百万个节点时，针对ConvGNNs的full-batch训练算法受到内存溢出问题的严重影响，最大的局限在于，无法动态的泛化新节点，只能用于静态图。

## 基于空间域的GCN的基本原理

<https://blog.csdn.net/yyl424525/article/details/103172141>

根据传统CNN在图像上的卷积操作，基于空间的GNN基于一个节点的空间关系定义图卷积算子。将图像看作特殊图形式，每个像素代表一个节点，如图1（a）所示，每个像素与附近的像素直接相连，如果用一个3×3窗口取块，每个节点的邻居节点就是其周围的八个像素，将滤波器作用于3×3块，则每个通道中心像素的值就是3×3块内像素的加权平均值。由于相邻结点有固定的顺序，所以可训练权重能够在不同的局部空间共享。如图1（b）所示，对于一般图结构，中心结点的表示也是根据其邻居结点的聚合结果表示。



a）2-D卷积。与图类似，将像素中的每个像素作为一个节点，像素的邻居节点由滤波器的大小决定。2-D卷积计算的是由红色节点和其邻居节点像素的加权平均值。节点的邻居都是有序的并且有固定大小

（b）图卷积。为了得到红色节点的隐式表示，图卷积算子的一个简单方法是取红色节点及其邻居节点的特征的平均值。与图像数据不同，节点的邻居是无序的且大小是可变的。

从另一个角度来看，基于空间的ConvGNNs与RecGNNs共享相同的信息传播/消息传递思想。空间图卷积运算实质上是沿着边缘传播节点信息。可以理解为消息传递式的操作。

消息式聚合**：**

随着图卷积越来越火，工业界逐渐加入了基础设施建设的队伍。借鉴 GraphX 等思路，出现一些不依赖邻接矩阵（或是屏蔽了邻接矩阵细节的）的消息聚合库，比较有名的有 PyG（比较早，实现多）和 DGL（比较新，易上手）。**在这些库中，节点可以发出信息，并接受周围节点的信息，显式地完成消息聚合**。在这种情况下，越来越多复杂的聚合方法出现了

<https://baijiahao.baidu.com/s?id=1619250802230567168&wfr=spider&for=pc>

## GraphSAGE

<https://cloud.tencent.com/developer/article/1489894>

在大规模图上学习节点 embedding ，在很多任务中非常有效，如学习节点拓扑结构的 DeepWalk 以及同时学习邻居特征和拓扑结构的 semi-GCN 。

但是现在大多数方法都是直推式学习， 不能直接泛化到未知节点。这些方法是在一个固定的图上直接学习每个节点 embedding ，但是大多情况图是会演化的，当网络结构改变以及新节点的出现，直推式学习需要重新训练（复杂度高且可能会导致 embedding 会偏移），很难落地在需要快速生成未知节点embedding的机器学习系统上。

众所周知，2017年ICLR出产的GCN现在是多么地热门，仿佛自己就是图神经网络的名片。然而，在GCN的风头中，很多人忽略了GCN本身的巨大局限——Transductive Learning——没法快速表示新节点，这限制了它在生产环境中应用。同年NIPS来了一篇使用Inductive Learning的GraphSAGE，解决了这个问题。今天，让我们来一起琢磨琢磨这个GraphSAGE是个什么玩意儿。

直推式 ( transductive ) 学习：从特殊到特殊，仅考虑当前数据。在图中学习目标是学习目标是直接生成当前节点的 embedding，例如 DeepWalk、LINE，把每个节点 embedding 作为参数，并通过 SGD 优化，又如 GCN，在训练过程中使用图的拉普拉斯矩阵进行计算。

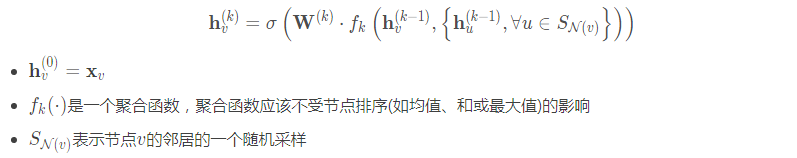
归纳 ( inductive ) 学习：平时所说的一般的机器学习任务，从特殊到一般：目标是在未知数据上也有区分性。

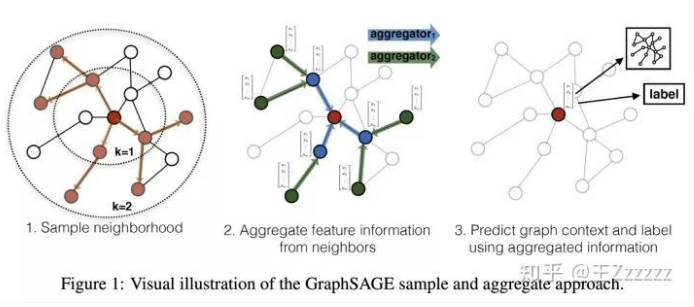
基本思想：

去学习一个节点的信息是怎么通过其邻居节点的特征聚合而来的。 学习到了这样的“聚合函数”，而我们本身就已知各个节点的特征和邻居关系，我们就可以很方便地得到一个新节点的表示了。

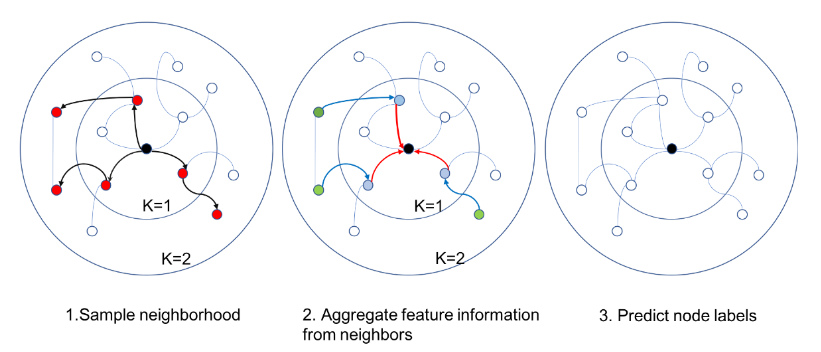
GCN等transductive的方法（****直推式 ( transductive ) 学习****），学到的是每个节点的一个唯一确定的embedding； 而GraphSAGE方法学到的node embedding，是根据node的邻居关系的变化而变化的，也就是说，即使是旧的node，如果建立了一些新的link，那么其对应的embedding也会变化，而且也很方便地学到。

聚合函数本质上是聚合节点的邻域信息，需要满足对节点顺序的排列保持不变，例如均值函数，求和函数，最大值函数都对节点的顺序没有要求。图的卷积运算定义为:





GraphSage的学习过程分为三个步骤。

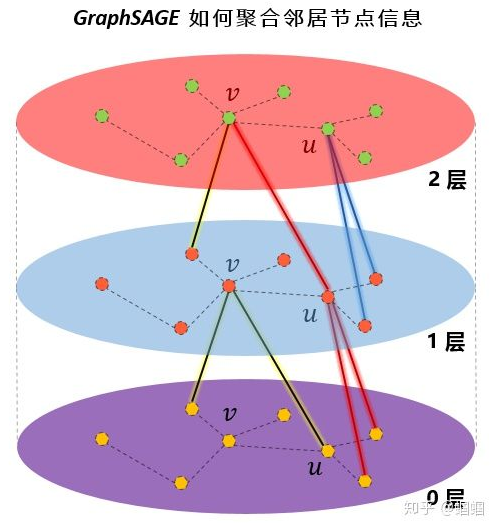


首先，采样，对一个节点的K-跳邻居节点取样。采中节点才参与特征聚合降低计算复杂度（图中一跳邻居采样数=3，二跳邻居采样数=5）设采样数量为k，若顶点邻居数少于k,则采用有放回的抽样方法，直到采样出k个顶点。若顶点邻居数大于k，则采用无放回的抽样。

然后，确定采样节点后，从最外层开始聚合特征，最终当前节点将聚合k层领域的信息。生成目标节点 emebedding：先聚合 2 跳邻居特征，生成一跳邻居 embedding ，再聚合一跳邻居embedding，生成目标节点embedding，从而获得二跳邻居信息。

与DeepWalk不同的是，这里的顶点表示向量是通过聚合顶点的邻接点特征产生的，而不是简单的进行一个embedding lookup操作得到。

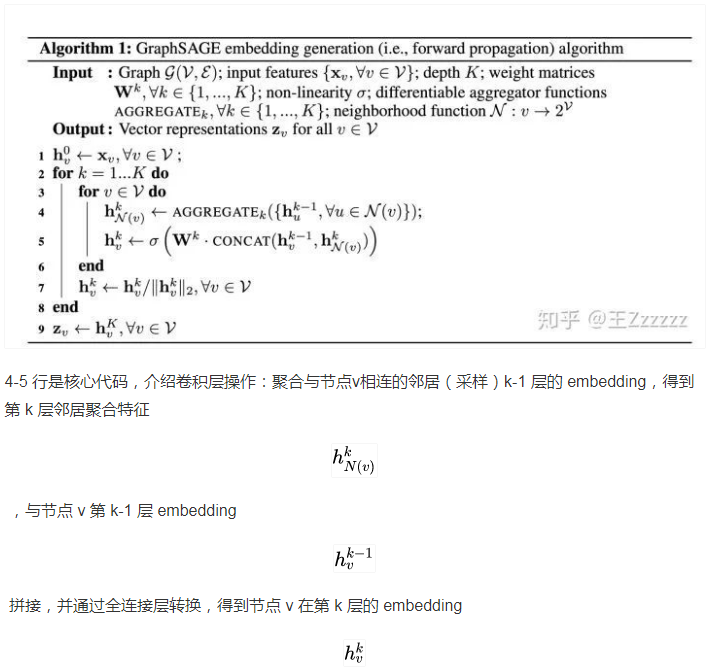
最后，将 embedding 作为全连接层的输入，利用最终状态做预测和误差反向传播。如图所示k-hop,从中心节点跳几步到达的顶点。



从0层到第二层，节点v已经聚合了其二阶领域的信息

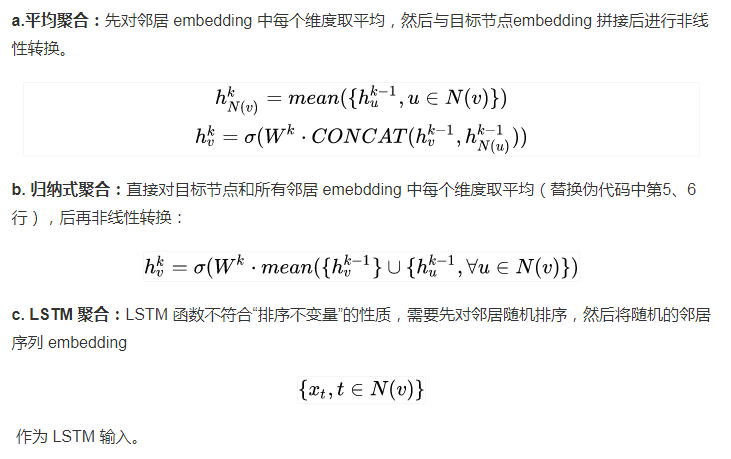
在GraphSAGE的实践中，作者发现，K不必取很大的值，当K=2时，效果就灰常好了，也就是只用扩展到2阶邻居即可。至于邻居的个数，文中提到S1×S2<=500，即两次扩展的邻居数之际小于500，大约每次只需要扩展20来个邻居即可。这也是合情合理，例如在现实生活中，对你影响最大就是亲朋好友，这些属于一阶邻居，然后可能你偶尔从他们口中听说一些他们的同事、朋友的一些故事，这些会对你产生一定的影响，这些人就属于二阶邻居。但是到了三阶，可能基本对你不会产生什么影响了，例如你听你同学说他同学听说她同学的什么事迹，是不是很绕口，绕口就对了，因为你基本不会听到这样的故事，你所接触到的、听到的、看到的，基本都在“二阶”的范围之内。

伪代码



聚合函数

伪代码第 5 行可以使用不同聚合函数，本小节介绍五种满足排序不变量的聚合函数：平均、GCN 归纳式、LSTM、pooling 聚合器。（因为邻居没有顺序，聚合函数需要满足排序不变量的特性，即输入顺序不会影响函数结果）聚合函数是GraphSage的精髓



## 基于谱和基于空间的模型的对比

谱模型是图数据处理的理论基础。基于谱的模型作为针对图数据最早期的卷积网络在很多图相关的分析任务种取得了非常好的效果。通过设计新的图信号滤波器(如Cayleynets)，理论上可以建立新的卷积神经网络。然而，由于效率、通用性和灵活性等问题，空间模型比谱模型更受欢迎。

## 效率

基于谱的方法的计算量会随着图的大小急剧增加，因为模型需要同时计算特征向量[[21]或者同时处理大图，这就使得模型很难对大图进行并行处理或缩放。基于空间的图方法由于直接对图域的邻居节点进行聚合，所以有潜力处理大图，方法是对一个batch数据计算而不是在整个图上计算。如果邻居节点的数量增加，能够通过采样技术GraphSage、LGCN[25,28] 提高效率。

## 通用性

基于谱的图方法假设图是固定的，因此对新的或者不同的图泛化性能很差。基于空间的方法在每个节点上进行局部图卷积，权值可以很容易地在不同地位置和结构之间共享。

## 灵活性

基于谱的模型只适用于无向图，谱方法用于有向图的唯一方法是将有向图转换为无向图（因为没有有向图的拉普拉斯矩阵明确的定义）。基于空间的模型可以将输入合并到聚合函数中，所以在处理多源输入像是边特征边方向上更灵活。

因此，近年来，基于空间的方法更受关注。

