我知道你们也没多少时间看了,所以我大概总结了一下:

第二点讲的是GAT(Garph Attention Networks)架构以及GAT的优点.第三点讲的是使用GAT进行实验,分为两类:半监督学习(Transductive Learning)与归纳学习(Inductive learning),其中Transductive Learning与半监督嵌入,流体正则化,Deep-Walk,ICA与Planetoid进行了比较,Inductive Learning与四个监督GraphSAGE归纳方法进行比较,主要就是那四个聚合器(GCN,mean,LSTM,pool).

2 GAT架构

2.1 图注意力层

2.1.1 输入与输出

图注意力层需要一个节点特征集h作为输入:



其中N为节点个数,F为每个节点的特征个数.

输出一个新的节点特征集h’:



按照节点输入特征预测节点输出特征,其中每个节点有F’个特征.

2.1.2 初始化步骤

为了有足够的表达能力对高层次的输入特征进行变换,至少需要一个可学习的线性变换.因此,一个通过权重矩阵W参数化的线性变换会应用到每一个节点:



这个矩阵表示输入节点的F个特征与输出节点F’个特征的关系.

2.1.3 self-attention与masked-attention

然后在节点上进行”自关注(self-attention)”,这个共同的注意力机制a为:



据此计算注意力相关系数:

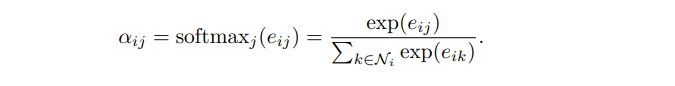


这个公式表示节点j的特征对节点i的特征的重要性,h就是前面的特征向量.

作者通过”相邻注意”(masked-attention),去把图注意力机制注入到图结构.这个”相邻注意”具体来说就是对于计算,只计算i的邻居节点j.

2.1.4 正则化

下一步是使用softmax函数进行正则化,用于对所有i的相邻节点j进行正则化:

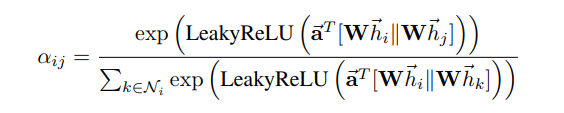


2.1.5 LeakyReLU激活

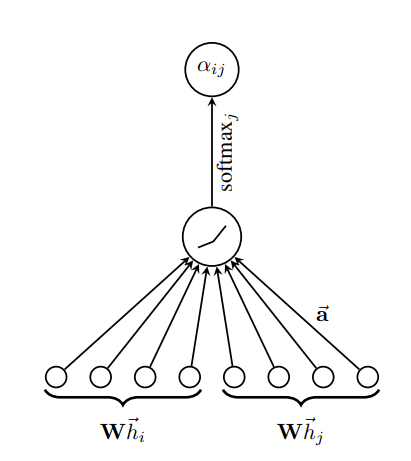
在这个实验中,注意力机制a是一个单层的前馈神经网络,通过一个权重向量:



进行参数化,并加入LeakyReLU非线性激活(输入的负斜率为0.2),然后就可以计算注意力相关系数了:



上标T表示转置,||表示串联运算(concatenation operation),图示如下:



四步:

1.权重矩阵W分别和i的特征hi与j的特征hj相乘

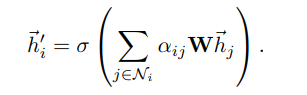
2.把结果串联在一起,再与相乘

3.经过LeakyReLU进行激活

4.使用softmax进行归一化得出注意力系数aij

2.1.6 计算输出特征

得到注意力系数后,通过如下公式计算输出特征:



其中:

(1)表示权值矩阵W与特征hj相乘的矩阵

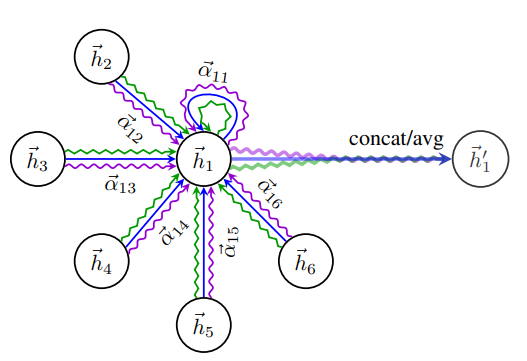
(2)为注意力系数

(3)表示遍历所有i的相邻节点j

(4) 为非线性激活函数

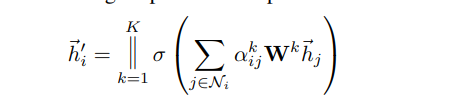
2.1.7 多端注意机制

“多端注意”(multi-head attention)机制用于稳定化”自注意”(self-attention)的学习过程,可以采用两种方式:串联或K平均.



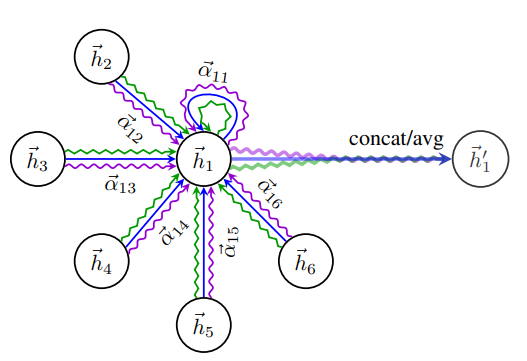
2.1.7.1 串联

把K个特征串联起来,输出特征如下:



||表示串联操作,是由第k个注意力机制计算出来的正则化注意力系数,一共有大K个”端”需要计算,为对应输入特征的线性变换权重矩阵,最终输出为h’,由KF’个特征组成(而不是原来的F’个特征).

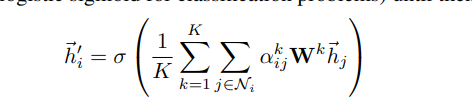
当K=3时,计算如下:



节点h1具有多端注意机制,不同风格颜色的箭头表示独立的注意力计算,从每个端聚合特征后串联(即图中的concat)得到h1’.

2.1.7.2 K平均

如果使用多端注意机制计算最终预测层的输入,串联操作可能不那么敏感,换句话说,可以使用均值化(K平均,即图中的avg)后再应用到最终的非线性函数(通常使用分类问题中的softmax或logistic sigmoid):



2.2. 优点

2.2.1 计算高效

时间复杂度为:



自注意力层可以并行计算每条边,可以并行计算每个节点得出输出特征.

2.2.2 更好鲁棒性

不同于GCNs,GAT能够对同一邻居的不同相邻节点给予不同的重要性.

2.2.3 不需要整张Graph

引入注意力机制后,只与相邻节点有关,与共享边节点无关

2.2.4 强于LSTM

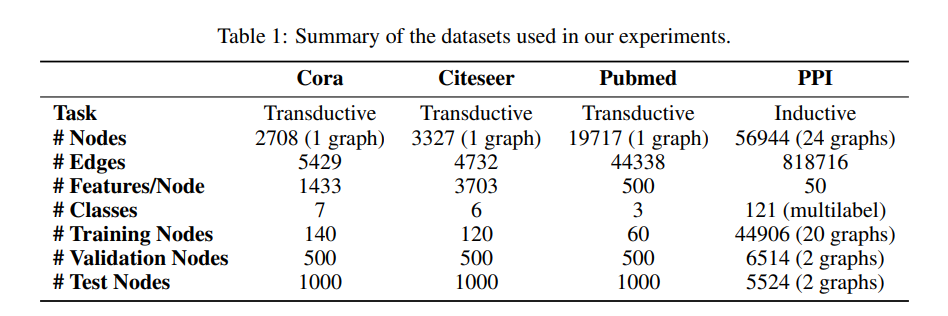
2.2.5 可以作为MoNet的一个实例

(优点这部分就不怎么翻译了…)

3 实验

实验分为两部分,半监督学习(Transductive Learning)与归纳学习(Inductive Learning).

3.1 数据集



其中Cora,Citeseer,Pubmed用于半监督学习,PPI用于归纳学习.

3.2 对比的方法

对于半监督学习,比较的方法有LP(label propagation),SemiEmb,(semi-supervised embedding,半监督嵌入),ManiReg(manifold regularization,流体正则化),Deep-Walk(skip-gram based graph embeddings,基于skip-gram的图嵌入),ICA(the iterative classification algorithm,迭代分类算法),Planetoid.同时直接把GAT模型与GCNs模型,高阶Chebyshev滤波器图卷积模型以及MoNet模型进行了比较.

对于归纳学习,比较的方法有四个不同的半监督GraphSAGE归纳方法:GraphSAGE-GCN(对归纳参数扩展了图卷积风格的操作),GraphSAGE-mean(计算特征向量的元素均值),GraphSAGE-LSTM(对邻域特征注入LSTM进行聚合)与GraphSAGE-pool(进行 共享非线性多层感知器转换的特征向量中的元素最大化操作).

3.3 实验建立

对于半监督学习:

- 应用了两层GAT模型

- 第一层包含K=8的多注意力端,去计算F’=8个特征(总共64个特征),ELU作为非线性激活函数(exponential linear unit)

- 第二层用于分类,一个注意力端计算C个特征(C为分类数目),后跟softmax激活.

- 为了应对小型训练集,训练过程中会应用的L2正则化. 的dropout会应用到这两层的输入以及归一化注意力系数.

对于归纳学习

- 应用了三层GAT模型

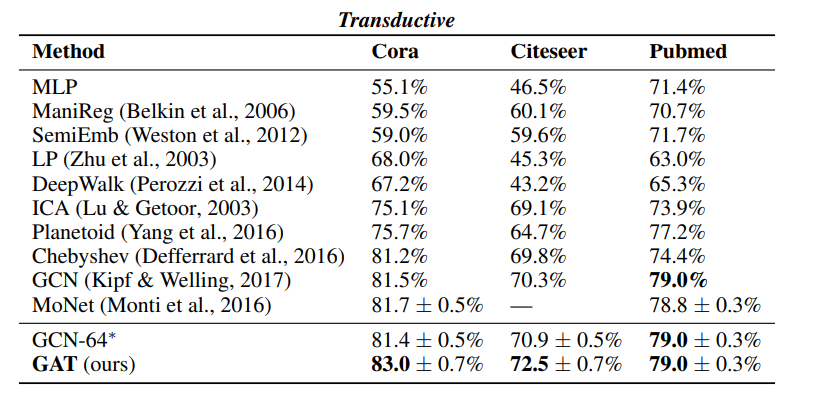
- 第一第二层包含K=4的多注意力端,去计算F’=256个特征(总共1024个特征),后跟ELU作为激活函数.

- 最后一层用于多标签分类,包含K=6的多注意力端,去计算F’=121个特征,激活函数为logistic sigmoid.

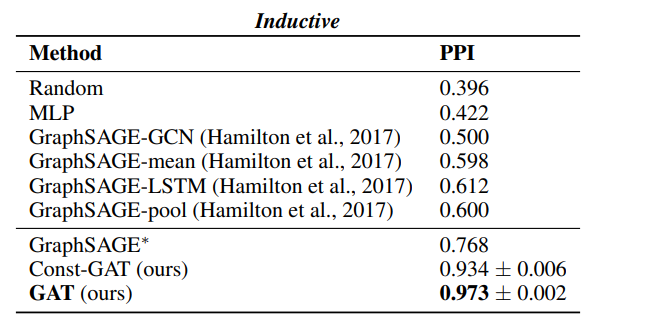
两个模型都使用Glorot初始化进行初始化,训练过程中使用Adam SGD优化器优化最小化交叉熵.

3.4 结果

半监督学习:



归纳学习:



GAT最厉害就对了….具体结果就不翻译了,看原文.

(偷偷给你们)几个问题(应该也许或许可能或者大概能在明天晚上提问):

1.文章使用了LeakyReLU,其中提到了负斜率,请问是什么意思?其他几种ReLU函数的斜率是怎么样的?

2.如何理解LeakyReLU中的串联操作?

3.正则化过程中用到了softmax函数,什么是softmax函数?

4.计算输出特征过程中使用的非线性激活函数可以是什么函数?

5.多端注意机制中的串联操作与K平均怎么理解?

6.建立实验的过程中用到了ELU函数,请问什么是ELU函数,对该实验的作用?

7.GAT的优点?

8.什么是L2-正则化? 如何影响L2正则化?

9.什么是Glorot初始化?什么是Adam SGD优化器?