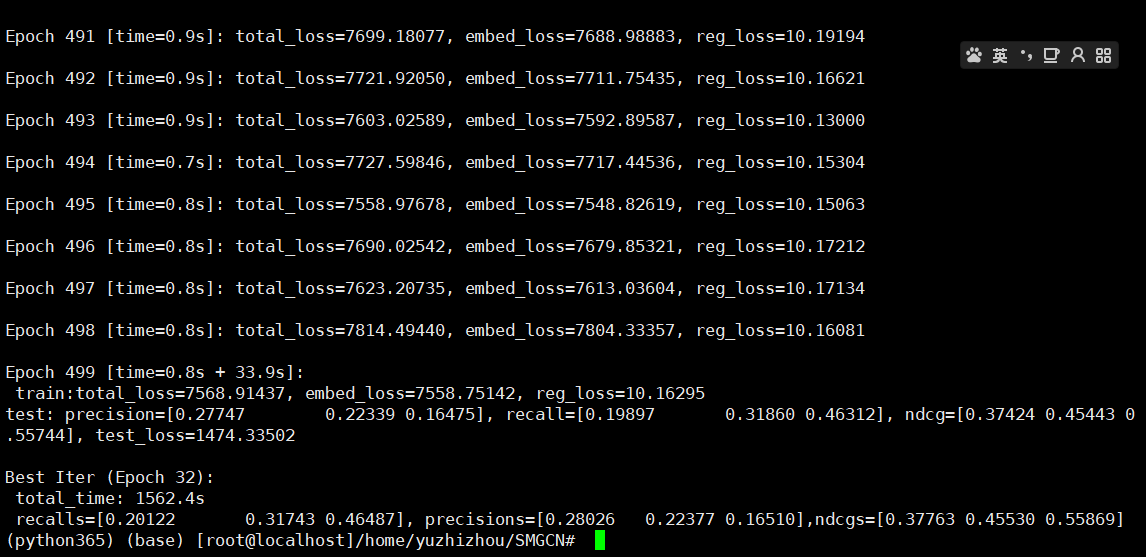
**学习周报**

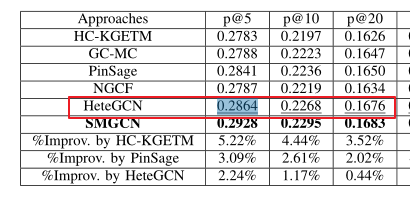
1. 学习内容与进度
2. 关于SGR代码的后续总结

|  |  |  |
| --- | --- | --- |
|  | 内容 | 总结 |
| 关于pytorch自定义模型中的自定义训练参数 |  | 单独使用定义模型属性的方法或者register\_parameter方法都是可以的，自定义的参数都会被自动放在模型的参数中；  最后一种方法是错误的，  出现的结果是grad =None并且模型参数也不会得到训练，原因是Varible变量不会被自动加入到模块参数中去，而且使用register\_parameter等价于定义了新的属性 |
| 关于分类层 | 之前在反卷积层之后添加了bn和relu层，  后来发现参考的代码都没有这样写，去掉后效果提高 | 加入激活函数是为了增加非线性，线性回归通常没有边界，可能连最简单的问题都拟合不了；  没有激活函数网络的深度没有意义 |
| 关于学长所说的在上采样后使用SGR | 在汇报前试过，但是显卡大幅度溢出，原因是数据复制：    尝试了两种优化策略，一种是尝试将一个向量和一个矩阵进行拼接，但是pytorch不支持；    第二种优化方式是直接使用数组来批量处理，虽然处理其中一张图片只需内存2.0MiB, 但是图片尺寸是200\*200=40000，仍然报错，所以只能计算一定数量的像素点：      缩小图像尺寸为150，依然显存不够；  一张显卡为11018MiB，所以只能处理11018/2=5509个像素点，  远小于150\*150=22500；要想让一张显卡能处理200\*200=40000个像素点，则处理每个像素的显存应该为11018/40000=0.27545，图片变得很小了。  也试过使用并行和清空GPU的显存缓存，都没有效果，并行容易出现一张显卡的显存不能满足基本的显存需求，就会导致无法并行 | 后面有时间试试GOogle colab，貌似其提供的显存要大很多 |
| 关于数据的目录结构和模型频繁变动版本 | 多个模型处理多个数据集，需要依据不同的迭代次数、模型、数据集的划分（train、val、test）、实验结果（模型检查点、测试集上预测结果和得分、日志）来设计目录；  对于不确定点，需要频繁变动模型，比如SGR中变动的有分组学习率和单一学习率、是否上采样再计算得分、每个epoch的每个iter是否需要每次backward还是所有iter结束再backward等 | 首先梳理清楚模型中可能需要经常变动的超参数以及实验各个环节的各种类别，将其设计为文件或者目录的一部分；  不要设置太多版本文件，做好改动记录 |

1. 完成毕设HetGCN代码

实现过程与SMGCN类似，只是对于邻居信息增加了注意力机制，准确率P@5为0.28026与论文结果0.2864相差0.006左右：





1. 学习PinSage和DGL库

PinSage是工业上首次将GCN用于大规模的推荐问题，由Pinterest公司提出并应用在pin推荐上。其以GraphSage算法为基础，但所有改进。GraphSage算法核心在于通过学习在图上生成节点表示的方法将GCN拓展成归纳学习任务，而非传统的直接学习节点的嵌入，其包含采样和聚合两个核心步骤。PinSage算法同样包含这两部分，在mini-batch采样部分采用基于随机游走的重要性采样，但非单纯的随机采样；在卷积层这一块相比GraphSage而言是一个多层的卷积结构，并且在邻居节点嵌入聚合前多增加了一层denselayer。

参考学习了tf实现的toy example，但是论文的很多细节都没有实现;于是又阅读学习了DGL库相关的API和文档，阅读DGL实现的PinSage，目前还有一部分没搞懂。

阅读记录直接整理在了代码注释中。

1. 学习问题

1、PinSage中用到了pandas处理数据，对这个库用的不多

1. 下周计划和安排
2. 初步阅读理解下上次选读的论文
3. 完成毕设PinSage的代码
4. 争取搭建好展示系统的界面