**1学习周报**

1. 学习内容与进度
2. 阅读理解PinSage结合dgl的源码

（1）源码可以分为几个核心步骤：

|  |  |
| --- | --- |
| 预处理数据 | 使用pandas来处理movielens1m数据集 |
| 利用dgl辅助构造双向的user-item二部图 | 包括计算两类节点的个数、’watched’以及’watched-by’两种关系对应的边、利用节点的相关属性来初始化ndata |
| 搭建采样模块 | 采样模块包含两个采样部分，都是为了构建训练时所需的数据加载器服务的。以DataLoader构建的过程来说明各个采样部分的作用：  【利用采样获取批量数据】：利用了ItemToItem采样模块，每次采样batch\_size个item节点、item节点对应的正负样本节点（正样本节点是item的在二部图上的两跳邻居、负样本节点使用了随机采样节点并非是原论文中的hard negative），也即heads、tails、neg\_tails ；  【处理获取的批量数据】：处理部分是为了从批量数据中构建相应的图结构，包括pos\_graph、neg\_graph、反映一阶邻居关系的二部图block（多个block反映出连续的一阶邻居关系）。pos\_graph  只包含heads和tails构成的边；neg\_graph只包含heads和neg\_tails构成的边；block结构即为利用dgl.to\_block函数将节点和其邻居的关联转换为二部图结构，相当于保存了部分图节点的邻居关系所构成的一个原图的子图，这里使用pos\_graph和neg\_graph中所有的  Item图节点来作为初始的seeds，然后连续采样多层并保存每次采样过程中的block结构，使用的采样方法是基于随机游走的重要性采样。 |
| 搭建PinSage模型 | 由线性映射器、多层卷积块、item间的交互损失计算这3个模块组成。  线性映射器将节点的不同属性特征映射为相同维度的特征表示，然后相加来代表该节点的嵌入；  多层卷积块利用之前保存的blocks结构在其上进行信息传播，得到更新后的嵌入；  损失计算分别计算pos\_graph和neg\_graph上的item嵌入内积的得分，然后利用正负样本损失函数来计算loss。 |

1. 常见的几点误区

|  |
| --- |
| Block的srcdata不是只包含了二部图的左边节点，而是包含了所有的节点 |
| Srcdata和dstdata在异质图中是不同的，在同质图中是相同的；  所以异质图上srcdata和dstdata不会相互影响 |
| 代码中的blocks数组是逆序插入单个block的，也就是说第一个block对应了多层采样后的所有节点，最后一个block对应了第一次采样后的所有节点 |

1. 实现毕设算法
2. 两点不同：PinSage是item到item的推荐，而这里是从sympt推荐herb；

PinSage使用正负样本损失，这里使用多标签损失

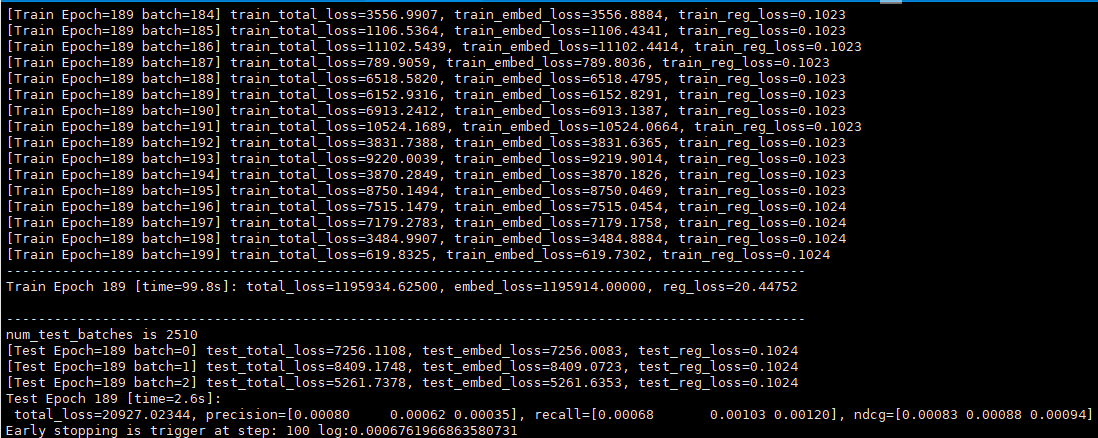
（2）使用tf实现：dgl要求tensorflow版本大于2.2, 即必须是eager\_execution模式

1. x代码中的placeholder在2.x中找不到替代函数，使用compat.v1又会关闭eager\_execution模式；

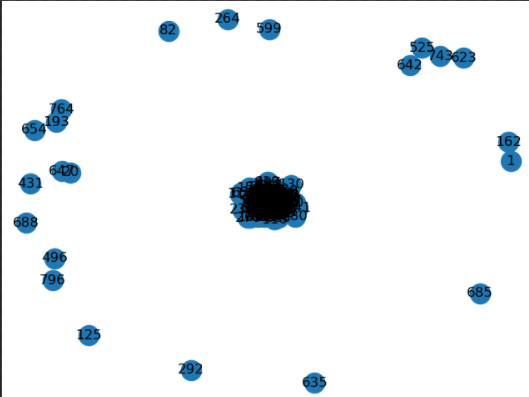
相互矛盾。

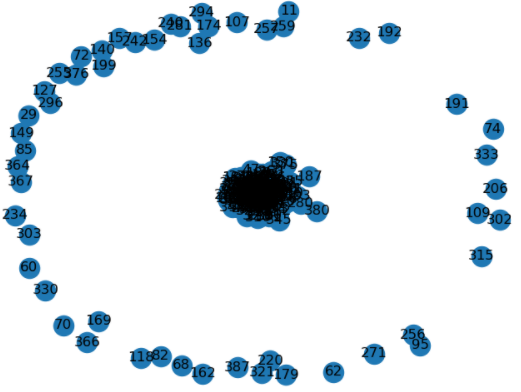
（3）使用pytorch：与dgl的结合不受版本限制

实验结果：准确率很低



可视化的同质图，异质图dgl不支持：





1. 完成毕设系统界面初步



1. 学习问题
2. 遇到的bug非常多，后面有时间再整理
3. 毕设比原先的预期时间要长