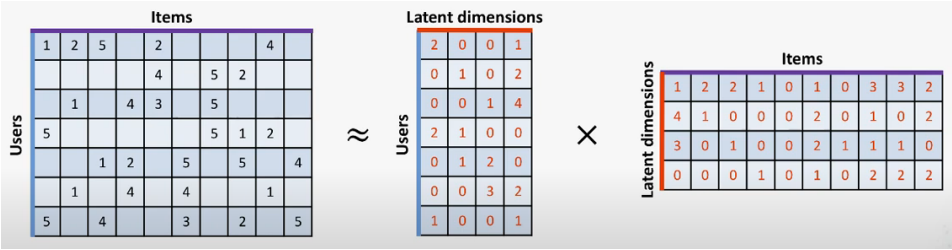
ICLR20一篇使用GNN来解决现有矩阵补全方法无法泛化问题的论文

解决的问题

如何让矩阵补全方法中一个数据集得到的embedding，能够迁移到另一个数据集上，同时不依赖额外的信息。

与另一篇论文GCMC应用的问题相同，具体地，这篇论文做的是推荐系统方向下的矩阵补全问题，给定一个评分矩阵，如何根据已有的评分记录来预测用户对其他物品的评分。传统的做法是将输入的评分矩阵分解成用户与物品的embedding，通过embedding重构评分矩阵，填补其中的缺失值，从而做出预测，如下图所示：



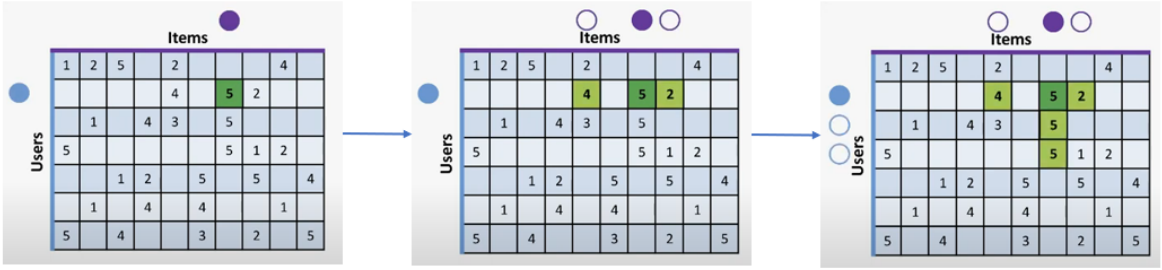
很多现有方法研究的都是如何得到更好的embedding，但它们都是直推式(transductive)而非启发式(inductive)的，意味着没法迁移，例如MovieLens数据集上得到的embedding就不能直接用于Douban数据集上，需要重新训练一个新的embedding。即使对于同一个数据集而言，如果加入新的评分记录，往往需要整个embedding重新训练。

做法及创新

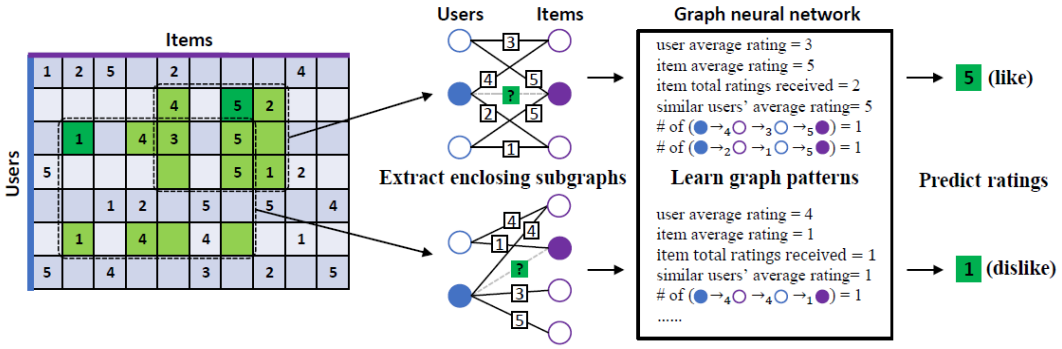
Enclosing Subgraph Extraction

论文的做法是为每一个评分记录提取一个子图，并且训练一个图神经网络来将得到的子图映射为预测评分。要想为评分记录提取子图，首先要将评分矩阵转换为图，转换的方法与另一篇论文GCMC相同，博客中有具体介绍，这里就不重复说明了。论文中对子图的定义方式为，给定一个评分记录(u, v)，表示用户u给物品v评过分，那么这个评分记录提取的子图由该用户u、物品v以及它们各自的h跳邻域内的顶点构成。为了具体说明是怎么从一个评分记录提取出子图的，我从论文作者的视频中截取了这部分内容，如下图所示：

假设第一张图中深绿色的方格是缺失值，这里先填入了模型的预测评分，倒退着来说明预测评分是怎么通过子图得到的。我们首先找到这个用户评过分的其它物品，对应于第五个物品的四分与第八个物品的两分，如第二张图所示。下一步是找到为这个物品评过分的其他用户，对应于第三个用户的五分与第四个用户的五分。



通过图二和图三找到的关系，就可以提取出这个评分记录的子图了，如下图所示：



可以看到，这个提取出的子图能提供许多有用的信息，例如用户平均评分、物品平均评分、物品累计评价次数以及基于路径的结构信息。论文希望通过这种结构信息来找到一些特征，从而做出预测，例如，如果用户u_0喜欢一个物品v_0，那么对于另一个与他品味相同的用户u_1，我们可能发现他也喜欢v_0。品味相同可以表示为两个用户都喜欢另一个物品v_1，这个特征可以表示为这么一条路径：$u_0 \to_{like} v_1 \to_{liked\ by} u_1 \to_{like} v_0$，如果$u_0$与$v_0$之间存在多条这样的路径，那么我们就可以推测$u_0$喜欢$v_0$。类似这样的结构特征数不胜数。因此，与其人工来手动定义大量这样的启发式特征(heuristics)，不如直接将子图输入一个图神经网络，来自动学习更通用的、更有表达能力的特征。

Node Labeling

这一步给顶点打标签是为了让子图中的顶点有着不同的角色，例如区分哪个是需要预测的目标用户与目标物品，区分用户顶点与物品顶点。而论文中打标签的方式十分简单：

- 目标用户与目标物品分别标记为0和1
- 对于h跳邻域内的顶点，如果是用户顶点标记为$2h$，物品顶点则标记为$2h+1$

标记之后，我们就能知道哪个是需要预测的目标用户与目标物品、哪些是用户顶点，因为用户顶点的标签均为偶数，以及邻域内顶点距离目标顶点距离的远近。这些标签将转换为one-hot编码的形式作为图神经网络输入的初始特征x_0。

这一节的最后论文作者还说到了这种标记方式与GCMC做法的不同之处。GCMC中同样是将标签转换为one-hot编码的形式作为GNN的初始特征，不同的是它用顶点在整个bipartite graph中的全局id作为它的标签，这等价于将GNN第一层信息传递网络的参数，转换为与每个顶点的全局id相关联的embedding函数，可以理解为一个embedding查找表，输入一个全局id，输出它对应的embedding。这显然是直推式的，对于不在查找表中的id，就无法得到它的embedding。这种情况对应于在小数据集上训练网络得到embedding，然后换到大数据集上，因为大数据集的顶点数量肯定要多于小数据集，这就会使得顶点的全局id范围变大，超出了训练出来的这个embedding查找表的范围。

Graph Neural Network

这一步的目的就是训练一个GNN来将提取出的子图映射成预测评分。论文所使用的GNN分为两个部分：信息传递层与池化层。前者的作用是得到子图中各顶点的特征向量，后者是根据得到的特征向量形成子图的一个特征表示。

信息传递部分使用的是R-GCN：

$$x_i^{l+1} = W_0^l x_i^l + \sum_{r \in R} \sum_{j \in N_r(i)} \frac{1}{|N_r(i)|} W_r^l x_j^l$$

其中x_i^l表示第i个顶点在第l层的特征向量，$N_r(i)$表示评分水平r下顶点i的邻域，顶点i以不同的边权重r所连接的顶点j用不同的参数矩阵W_r^l来进行处理。通过堆叠L层网络可以得到顶点i的L个特征向量，通过拼接的方式得到它最终的特征表示h_i：

$$h_i = \text{concat}(x_i^1, x_i^2, \ldots, x_i^L)$$

池化部分只选取子图中目标用户与目标顶点的特征向量进行拼接，来得到该子图的特征表示，这么做的原因是这两个顶点携带了最多的信息。

$$g = \text{concat}(h_u, h_v)$$

在得到子图的特征表示后，最后一步是通过一个MLP将它转换为一个预测评分$\hat{r}$：

$$\hat{r} = w^T \sigma(Wg)$$

Adjacent Rating Regularization

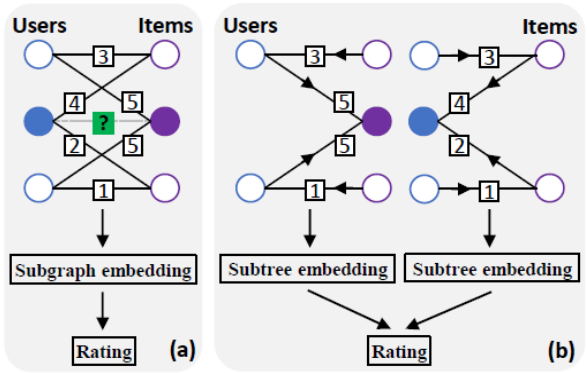
论文对于信息传递部分使用的R-GCN还提出了一点改进，在原始的R-GCN中，不同的评分水平是独立看待的，彼此之间没有关联，例如对于1、4、5这三个评分，显然地4和5都表示了用户的喜爱而1表示了用户的厌恶，同时4和5的相似程度要大于4和1，但这种次序关系及大小关系在原始的R-GCN中都被丢掉了。因此本论文添加了一个约束来引入这部分丢失的信息，具体做法也很简单，就是使得相邻的评分水平使用的参数矩阵更加相似：

$$L_{ARR} = \sum_{i=1,2,\ldots,|R|-1} ||W_{r_i+1} - W_{r_i}||_F^2$$

这里假设评分$r_1, r_2, \ldots, r_{|R|}$表示了用户喜爱程度的递增，通过这个约束就保留了评分的次序信息，同时可以使得出现次数较少的评分水平可以从相邻的评分水平中迁移信息，来弥补数据不足带来的问题。

Graph-level GNN vs Node-level GNN

这一节还是在于GCMC作比较。在GCMC中，采用的是顶点层面的图神经网络，它应用于图中的顶点来得到顶点的embedding，再通过embedding得到预测评分，如下右图所示。这么做的缺陷是它独立地学习两个顶点所关联的子树，而忽略了这两棵子树之间可能存在的联系。



数据集

Flixster、Douban、YahooMusic、ML-100K、ML-1M