RDASGC: a more powerful GNN architecture for recommendation

1. Introduction

NGCF[1]将微调后的 GCN[2]模型应用到推荐系统，通过 embedding propagation 显示学习协同信号，在多个数据集上取得了 SOTA 的效果。然而，GNN 的多项研究成果表明[3], [4], [5]，transformation 和 propagation 的耦合不仅会增加模型的训练难度，也不利于模型的表达能力。基于 GNN 的以上研究进展，我们设计了一个更高效的 GNN 模型结构，主要包括 neighbor aggregation 和 layer aggregation 两部分。实验结果表明，在相同的设置下，我们的模型在多个数据集上的表现均显著超越了 NGCF，提升幅度最高可达 **22%**。代码已上传至 github，数据集也包括在内，https://github.com/lt610/RDASGC。

2. Baseline

我们对比的 baseline 为 NGCF，该模型已经在多个数据集上取得了 SOTA 的效果，论文中的详细结果见 Table 1。可以看到，NGCF 的表现不仅超越了 MF 等传统模型，也超过了 GC-MC 等 GNN 模型。

Table 1: NGCF's results

	Gowalla		Yelp2018*		Amazon-Book	
	recall	ndcg	recall	ndcg	recall	ndcg
MF	0.1291	0.1109	0.0433	0.0354	0.0250	0.0196
NeuMF	0.1399	0.1212	0.0451	0.0363	0.0258	0.0200
CMN	0.1405	0.1221	0.0457	0.0369	0.0267	0.0218
HOP-Rec	0.1399	0.1214	0.0517	0.0428	0.0309	0.0232
GC-MC	0.1395	0.1204	0.0462	0.0379	0.0288	0.0224
PinSage	0.1380	0.1196	0.0471	0.0393	0.0282	0.0219
NGCF-3	**0.1569***	**0.1327***	**0.0579***	**0.0477***	**0.0337***	**0.0261***
%Improv.	11.68%	8.64%	11.97%	11.29%	9.61%	12.50%
p-value	2.01e-7	3.03e-3	5.34e-3	4.62e-4	3.48e-5	1.26e-4

NGCF 的原理见公式(1)-(2)。该模型在 GCN 的基础上做了一些微调，加入了亲和项使得相似的 item 可以传播更多信息，同时将 ReLU 激活函数替换成了 LeakyRELU。

$$E^{(l)} = LeakyReLU\left((\mathcal{L}+I)E^{(l-1)}W_1^{(l)} + \mathcal{L}E^{(l-1)} \odot E^{(l-1)}W_2^{(l)}\right) \tag{1}$$

$$\mathcal{L} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}} and A = \begin{bmatrix} 0 & R \\ R^T & 0 \end{bmatrix} \tag{2}$$

在预测时，将每一层的输出 concatenate 后作为最终的 embedding，然后通过内积运算计算 user 对 item 的偏好评分，见公式(3)-(4)所示。

$$e_u^* = e_u^{(0)}||\dots||e_u^{(l)}, e_i^* = e_i^{(0)}||\dots||e_i^{(l)} \tag{3}$$

$$\hat{y}NGCF(u,i) = e_u^{*T} e_i^* \tag{4}$$

损失函数采用 BPRLoss，并加入惩罚项防止过拟合，见公式(5)所示。

$$Loss = \sum_{(u,i,j)\in O} -ln\sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda||\Theta||_2^2 \tag{5}$$

3. Models

我们的模型（RDASGC）参考了 SGC[3]，PPNP[4]，JKNet[6]，GRAND[7]和 DAGNN[5]，主要分为 neighbor aggregation 和 layer aggregation 两部分。其中 neighbor aggregation 去除了冗余的 transformation 和 nonlinearity，用于显示编码协同信息；layer aggregation 分为 PPR、Adaptive、Mean、1/K 四种方式，用于学习局部和全局信息。模型的整体结构见 Figure 1 所示

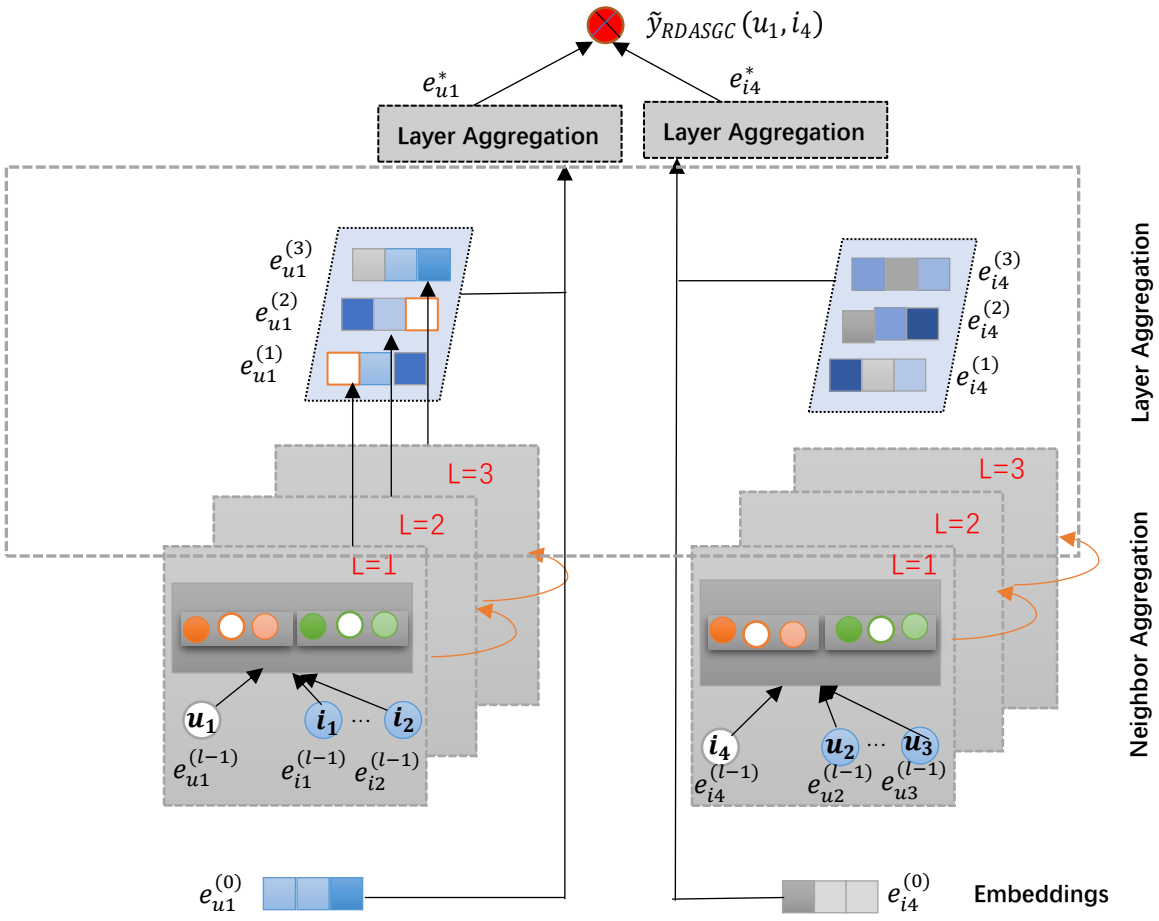

Figure 1 Architecture of RDASGC

PPR layer aggregation 是 APPNP 的采用的传播方式，见公式(6)所示。

$$\mathrm{E}^{(l+1)} = \alpha \mathrm{E}^{(0)} + (1-\alpha)\tilde{A}\mathrm{E}^{(l)} \tag{6}$$

将公式(6)展开可以得到公式(7)，相当于对每一层的输出进行加权求和，加权系数随层数指数衰减，这意味着近距离的邻居更重要，我们可以通过α来平衡该重要性。

$$\begin{aligned}\mathrm{E}^{(l)} &= \alpha \mathrm{E}^{(0)} + (1-\alpha)\tilde{A}\mathrm{E}^{(l-1)} \\ &= \alpha E^{(0)} + \alpha\tilde{A}E^{(0)} + \alpha(1-\alpha)^2\tilde{A}^2E^{(0)} + \cdots + (1-\alpha)^l\tilde{A}^lE^{(0)}\end{aligned} \tag{7}$$

Adaptive layer aggregation 可以为每个结点分别学习不同范围邻居信息的重要性，相当于 PPR 的增强版本，见公式(8)-(12)所示。

$$E^{(l)} = \tilde{A}^l Z, l = 1,2,\dots,L \tag{8}$$

$$E = stack\left(Z, E^{(1)}, \dots, E^{(l)}\right) \tag{9}$$

$$S = \sigma(Es) \tag{10}$$

$$\tilde{S} = reshape(S) \tag{11}$$

$$X_{out} = softmax\left(squeeze\left(\tilde{S}E\right)\right) \tag{12}$$

Mean 和 1/K 是 layer aggregation 的简化情形，分别假设了不同范围的邻居结点同样重要，以及近距离的邻居更重要并且重要性以 1/K 递减。

4. Experiment

4.1 Dataset

实验用到了三个数据集，规模从 100w 到 300w，详细信息见 Table 1 所示。为了保证数据集的质量，原文使用了 10-core 设置，即保留用户至少 10 次的交互。Gowalla 是基于地理位置用户兴趣推荐的签到数据集；Yelp2018 是 Yelp 2018 底特律餐厅推荐比赛的数据集；Amazon-book 是 Amazon-reviews 产品推荐数据集的一个子集。

Table 2: Statistics of datasets

Dataset	User #	Item #	Interaction#	Density
Gowalla	29,858	40,981	1,027,370	0.00084
Yelp2018*	31,668	38,048	1,561,406	0.00130
Amazon-Book	52,643	91,599	2,984,108	0.00062

4.2 Implementation

我们基于 NGCF 的代码（TensorFlow 实现[8]）采用 PyTorch 和 DGL 实现了文中的模型。DGL（Deep Graph Learning）是 AWS 的一款面向图神经网络的开源框架。实验环境为 Python 3.7.0、PyTorch 1.7.0、DGL 0.5.2、CUDA 10.2、GeForce GTX 1080 Ti 和 GeForce GTX 2080 Ti。

4.3 Setting

训练集和测试集的划分与[1]一致。在训练集中随机选取 10%的交互关系作为验证集，用于超参数的调优。评价指标采用 recall@K 和 ndcg@K，K=20。使用 early stopping，50 个 epochs 内验证集上的 recall@20 不再提高就停止训练。Gowalla 和 Yelp2018 上的 batch size 设置为 2048，Amazon-book 上的 batch size 设置为 8192。使用 grid search，k: {1, 2, 3, 4}，learn rate: {0.0001, 0.0005, 0.001}，weight deacy: {1e-4, 5e-4, 1e-3}，alpha: {0.1, 0.05}。

4.4 Result

A. Effect of layer aggregation

Table 3: Results of different layer aggregation on Gowalla

Layer Aggregation	None	Mean	1/K	PPR	Adaptive
recall	0.1749	**0.1826**	0.1761	0.1455	0.1792
ndcg	0.1459	**0.1551**	0.1510	0.1126	0.1507

首先我们在 Gowalla 数据上对不同的 layer aggregation 进行了比较实验。可以看到，即使不使用任何 layer aggregation，我们的模型也比 NGCF 的表现要好。PPR 的表现是最差的，这是因为 PPR 对超参数α比较敏感，由于时间限制我们并没有精调α。

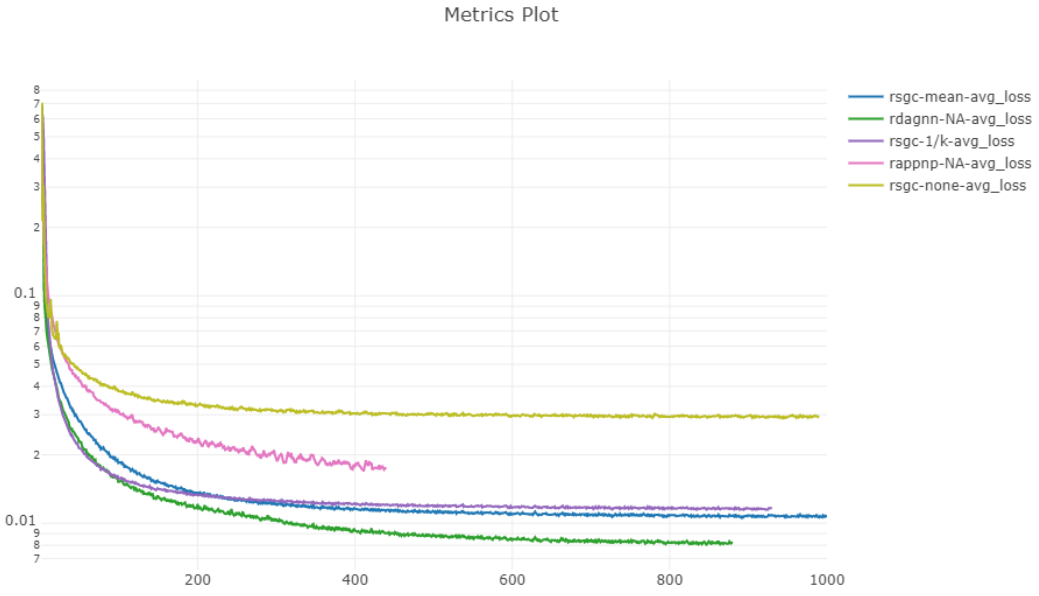

Figure 2: Loss trend of different layer aggregation

另外值得注意的是，最简单的 mean 的表现是最好的，甚至超过了我们寄予厚望的 Adaptive。为此我们将训练过程进行了可视化，loss 随 epoch 的变化

曲线见 Figure 2 所示。可以看到，Adaptive（绿色的曲线）的拟合能力是最强的。

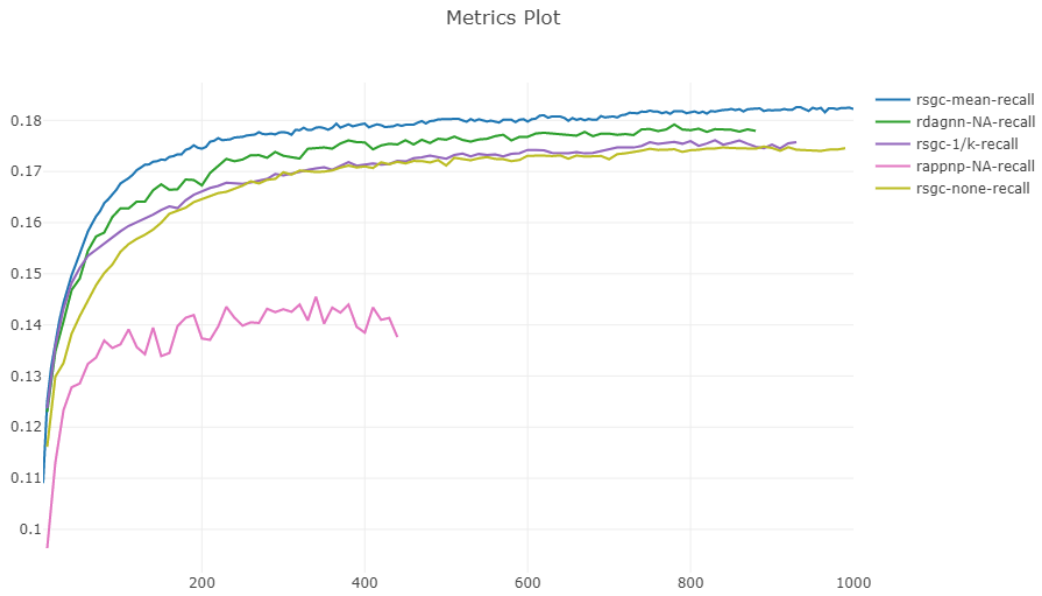

Figure 3: recall trend of different layer aggregation

Recall 随 epoch 的变化曲线见 Figure 3 所示，我们可以看到 Adaptive 的表现一直比 mean 差，NDCG 也有相似的情况。我们分析这很可能是因为 Adaptive 的学习能力比较强，然而在实验中我们并没有采用 Dropout、DropEdge、DropNode、L_2等正则化方法，因此发生了过拟合。

B. Overall Comparison

Table 3 Overall Results

Models	Gowalla		Yelp2018		Amazon-Book	
	recall	ndcg	recall	ndcg	recall	ndcg
MF	0.1291	0.1109	0.0433	0.0354	0.0250	0.0196
NeuMF	0.1399	0.1212	0.0451	0.0363	0.0258	0.0200
CMN	0.1405	0.1221	0.0457	0.0369	0.0267	0.0218
HOP-Rec	0.1399	0.1214	0.0517	0.0428	0.0309	0.0232
GC-MC	0.1395	0.1204	0.0462	0.0379	0.0288	0.0224
PinSage	0.1380	0.1196	0.0471	0.0393	0.0282	0.0219
NGCF	0.1569	0.1327	0.0579	0.0477	0.0377	0.0261
RDASGC-mean	**0.1826**	**0.1551**	**0.0643**	**0.0524**	**0.0416**	**0.0320**
%Improv.	**16.38%**	**16.88%**	**11.05%**	**9.85%**	**10.34%**	**22.61%**

接着我们用表现最好的 RDASGC-mean 在三个数据集上进行了调参，最终结果见 Table 3。可以看到，我们的模型的表现大幅度超越了 NGCF，最高可达到 22%。

C. Effect of layer numbers

Table 4: Effect of layer numbers

Layer Numbers		None	Mean	1/K	PPR	Adaptive
1	recall	0.1744	0.1590	0.1722	0.1206	0.1746
	ndcg	**0.1467**	0.1358	0.1469	0.0931	0.1489
2	recall	**0.1749**	0.1770	0.1741	0.1381	0.1761
	ndcg	0.1459	0.1509	0.1491	0.1102	0.1498
3	recall	0.1557	**0.1826**	0.1741	0.1413	0.1761
	ndcg	0.1245	**0.1551**	0.1490	0.1105	0.1490
4	recall	0.0925	0.1823	**0.1761**	**0.1455**	**0.1792**
	ndcg	0.0652	0.1537	**0.1510**	**0.1126**	**0.1508**

最后我们实验分析了层数的变化对模型表现的影响，实验结果见 Table 4。可以看到，如果不采用任何 layer aggregation，当层数增加到 3 层时，模型性能就会大幅下降，这一现象在 GNN 领域也被称为过平滑。而采用了 layer aggregation 后，层数的增加却可以带来性能上的提升。

5. Explanation

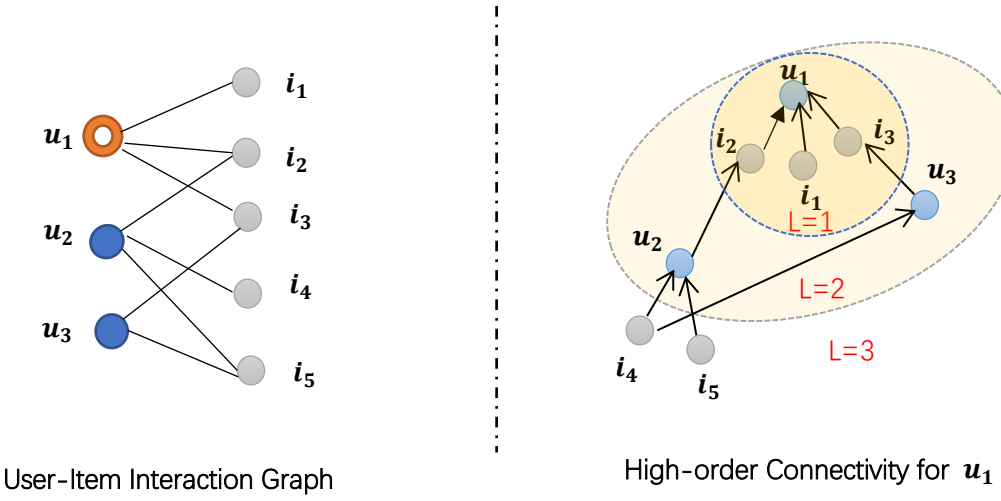

Figure 4

最后是模型的可解释性部分。如 Figure 4 所示，左图是用户-商品的二分关系图，右图是用户u_1的三阶邻居结构图。它很好地解释了为什么 GNN 能够有

效地学习到协同信息。例如，用户u_1和用户u_2都购买了商品i_2，用户u_2又购买了商品i_4和i_5，因此用户u_1也有可能够买商品i_4和i_5。此外，又因为还存在路径$u_1 - i_3 - u_3 - i_4$，所以相比商品i_5，用户u_1更有可能够买商品i_4。这样通过 neighbor aggregation 平滑操作后，用户u_1和商品i_4的 embedding 会更加接近，通过内积计算得到的偏好得分也会更高，在 topk 推荐中就会位于排名更加靠前的位置。

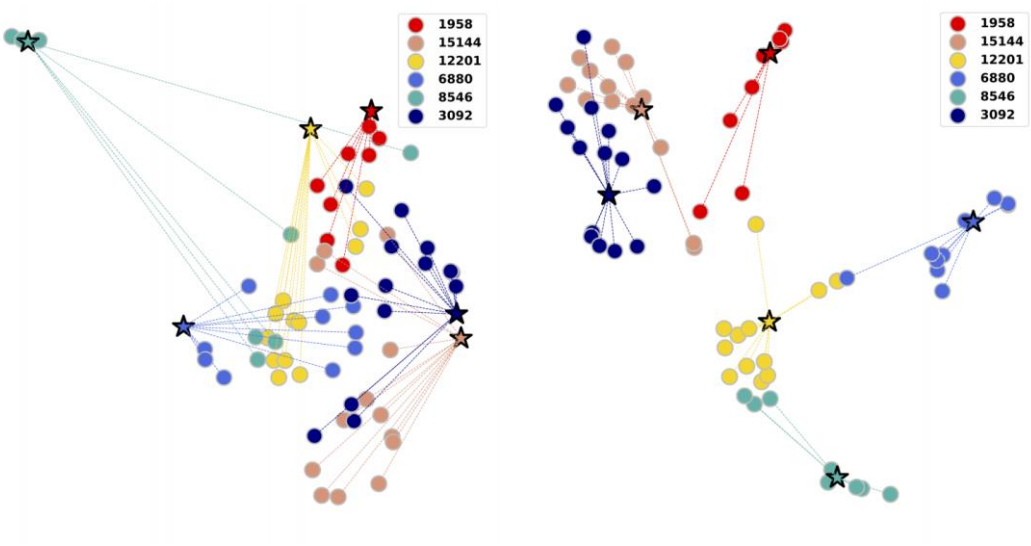

Figure 5: Visualization of the learned t-SNE transformed representations derived from MF and RDASGC-3. Each star represents a user from Gowalla dataset, while the points with the same color denote the relevant items

为了验证以上分析，我们对学习到的 embedding 进行了可视化。左图是 MF 的结果，相当于去掉了我们模型中的 neighbor aggregation 和 layer aggregation。右图是我们模型的结果，可以看到，用户和与其相关的商品，以及具有相似行为的用户及其相关商品，他们的 embedding 经过平滑后更接近了，这充分证实了以上分析。

6. 小组分工

代码主要封装设计为以下部分：数据处理、损失函数、评价指标、训练过程、测试过程、模型

刘唐 20210980061：负责文献的搜集和主题的确定，模型、训练和测试过程的实现，实验的调参分析，部分报告的编写，PPT 的编写

周彦 20210980065：负责损失函数和评价指标的实现，实验数据的可视化，部分报告的编写

张逸敏 20210980072：负责数据处理部分的实现、实验数据的整理，部分报告的编写

References

[1] Wang X, He X, Wang M, et al. Neural graph collaborative filtering[C]//Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval. 2019: 165-174.

[2] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arXiv preprint arXiv:1609.02907, 2016.

[3] Wu F, Zhang T, Souza Jr A H, et al. Simplifying graph convolutional networks[J]. arXiv preprint arXiv:1902.07153, 2019.

[4] Klicpera J, Bojchevski A, Günnemann S. Predict then propagate: Graph neural networks meet personalized pagerank[J]. arXiv preprint arXiv:1810.05997, 2018.

[5] Liu M, Gao H, Ji S. Towards deeper graph neural networks[C]//Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2020: 338-348.

[6] Xu K, Li C, Tian Y, et al. Representation learning on graphs with jumping knowledge networks[J]. arXiv preprint arXiv:1806.03536, 2018.

[7] Feng W, Zhang J, Dong Y, et al. Graph Random Neural Networks for Semi-Supervised Learning on Graphs[J]. Advances in Neural Information Processing Systems, 2020, 33.

[8] https://github.com/xiangwang1223/neural_graph_collaborative_filtering