華東師範大學
EAST CHINA NORMAL UNIVERSITY

GNN可解释性+Recommender Systems

汇报人：岳文静

2022年5月17日

《MixGCF: An Improved Training Method for Graph Neural Network-based Recommender Systems》

By：KDD 2021
清华大学（唐杰团队）
& 浙江大学
& Facebook

背景

- 数据集
  - 显式数据集：能明显反映用户对物品的喜爱程度（如用户对电影的评分），但不易收集。
  - 隐式数据集：用户的行为数据（如用户的浏览、收藏、购买行为），是推荐系统中常见的数据，只能从这些行为数据去推测，用户是否对物品的喜爱。
- 负反馈：数据集中的负样本信息（显式数据集中的低评分，隐式数据集中与用户没有交互过的行为信息）。
- 难负例：在隐式数据集中，所有交互过的行为统一视为正例，没有发生过交互的物品可以采样出一部分作为负例，和正例相似的负样本称为难负例。

动机

- 隐式负反馈
  - GNN是CF(协同过滤) 的一个SOTA解决方案;
  - 学习隐式负反馈信息是CF的一大挑战;
  - 基于GNN的CF模型中的负采样方法未被探索;
- 难负例采样
  - 已有的难负例采样方法[*,**]均基于原始样本数据进行采样。

* Zhen Yang, Ming Ding, Chang Zhou, Hongxia Yang, Jingren Zhou, and Jie Tang. 2020. Understanding Negative Sampling in Graph Representation Learning. In KDD.

** Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In KDD. 974–983.

贡献

- 提出了一个负样本采样方法，不是直接从原始样本中进行负采样，而是通过生成的方式进行负采样，以改进基于GNN的推荐系统。

- 提出了一个即插即用的MixGCF框架，其中包含positive mixing和hop mixing策略，可以自然地将负样本的采样融入基于GNN的推荐模型中。

MixGCF

Algorithm 1: The training process with MixGCF

Input: Training set $\{(u, v^+)\}$, Recommender f_{GNN}, Number of negative candidate M, Number of aggregation layers L.

for $t = 1, 2, \cdots, T$ **do**

  Sample a mini-batch of positive pairs $\{(u, v^+)\}$.

  Initialize loss $\mathcal{L} = 0$.

  // Negative Sampling via MixGCF.

  for *each* (u, v^+) *pair* **do**

    Get the aggregated embeddings of each node by f_{GNN}.

    Get the set of candidate negative embeddings $\mathcal{E}$ by uniformly sampling M negatives.

    Get the updated set of negative candidate $\mathcal{E}'$ by (5).

    Synthesize a hard negative $\mathbf{e}_{v^-}$ based on $\mathcal{E}'$ by (6).

    $\mathcal{L} = \mathcal{L} + \ln \sigma(\mathbf{e}_u \cdot \mathbf{e}_{v^-} - \mathbf{e}_u \cdot \mathbf{e}_{v^+})$.

  end

  Update θ by descending the gradients $\nabla_\theta \mathcal{L}$.

end

Positive Mixing

通过在负样本中加入正样本的信息，从而使得生成的新的负样本更靠近正样本，作为正例增强后的负例候选集。

Hop Mixing

基于负例候选集和GNN的聚合过程，生成最终的难负例。

MixGCF

Algorithm 1: The training process with MixGCF

Input: Training set $\{(u, v^+)\}$, Recommender f_{GNN}, Number of negative candidate M, Number of aggregation layers L.

for $t = 1, 2, \cdots, T$ **do**
- Sample a mini-batch of positive pairs $\{(u, v^+)\}$.
- Initialize loss $\mathcal{L} = 0$.
- // Negative Sampling via MixGCF.
- **for** *each* (u, v^+) *pair* **do**
  - Get the aggregated embeddings of each node by f_{GNN}.
  - Get the set of candidate negative embeddings $\mathcal{E}$ by uniformly sampling M negatives.
  - Get the updated set of negative candidate $\mathcal{E}'$ by (5).
  - Synthesize a hard negative $\mathbf{e}_{v^-}$ based on $\mathcal{E}'$ by (6).
  - $\mathcal{L} = \mathcal{L} + \ln \sigma(\mathbf{e}_u \cdot \mathbf{e}_{v^-} - \mathbf{e}_u \cdot \mathbf{e}_{v^+})$.
- **end**
- Update θ by descending the gradients $\nabla_\theta \mathcal{L}$.

end

Positive Mixing

step1：得到与用户u相关的item正例v^+M个item负例v^-在GNN中每一层 (l) 的嵌入表示集合：

$$\mathcal{E} = \{\mathbf{e}_{v_m}^{(l)}\} \text{ of size } M \times (L+1).$$

step2：以一定的比例$\alpha^{(l)}$注入正例的信息，更新负例的表示, 其中，$\alpha^{(l)} \in U(0,1)$。

$$\mathbf{e'}_{v_m}^{(l)} = \alpha^{(l)}\mathbf{e}_{v^+}^{(l)} + (1 - \alpha^{(l)})\mathbf{e}_{v_m}^{(l)}, \alpha^{(l)} \in (0, 1), \qquad (5)$$

step3：得到正例增强后的负例候选集

$$\mathcal{E}' = \{\mathbf{e'}_{v_m}^{(l)}\}$$

MixGCF

Algorithm 1: The training process with MixGCF

Input: Training set $\{(u, v^+)\}$, Recommender f_{GNN}, Number of negative candidate M, Number of aggregation layers L.

for $t = 1, 2, \cdots, T$ **do**
- Sample a mini-batch of positive pairs $\{(u, v^+)\}$.
- Initialize loss $\mathcal{L} = 0$.
- // Negative Sampling via MixGCF.
- **for** *each* (u, v^+) *pair* **do**
  - Get the aggregated embeddings of each node by f_{GNN}.
  - Get the set of candidate negative embeddings $\mathcal{E}$ by uniformly sampling M negatives.
  - Get the updated set of negative candidate $\mathcal{E}'$ by (5).
  - Synthesize a hard negative $\mathbf{e}_{v^-}$ based on $\mathcal{E}'$ by (6).
  - $\mathcal{L} = \mathcal{L} + \ln \sigma(\mathbf{e}_u \cdot \mathbf{e}_{v^-} - \mathbf{e}_u \cdot \mathbf{e}_{v^+})$.
- **end**
- Update θ by descending the gradients $\nabla_\theta \mathcal{L}$.

end

Hop Mixing

step1：在每层随机选择一个负样本。

$$\mathbf{e'}^{(l)}_{v_x} = \mathop{\arg\max}_{\mathbf{e'}^{(l)}_{v_m} \in \mathcal{E}^{(l)}} f_{\mathrm{Q}}(u, l) \cdot \mathbf{e'}^{(l)}_{v_m}, \qquad (8)$$

$$\mathbb{E}\left[||(\theta_T - \theta^*)_u||^2\right] = \frac{1}{T}\left(\frac{1}{p_d(v|u)} - 1 + \frac{1}{Kp_n(v|u)} - \frac{1}{K}\right), \qquad (7)$$

step2：通过GNN的池化操作对这些选择出来的负样本embedding进行聚合。

$$\mathbf{e}_{v^-} = f_{\text{pool}}\left(\mathbf{e'}^{(0)}_{v_x}, \cdots, \mathbf{e'}^{(L)}_{v_y}\right), \qquad (6)$$

Zhen Yang, Ming Ding, Chang Zhou, Hongxia Yang, Jingren Zhou, and Jie Tang. 2020. Understanding Negative Sampling in Graph Representation Learning. In KDD.

Algorithm 1: The training process with MixGCF

Input: Training set $\{(u, v^+)\}$, Recommender f_{GNN}, Number of negative candidate M, Number of aggregation layers L.

for $t = 1, 2, \cdots, T$ **do**

- Sample a mini-batch of positive pairs $\{(u, v^+)\}$.
- Initialize loss $\mathcal{L} = 0$.
- // Negative Sampling via MixGCF.
- **for** *each* (u, v^+) *pair* **do**
  - Get the aggregated embeddings of each node by f_{GNN}.
  - Get the set of candidate negative embeddings $\mathcal{E}$ by uniformly sampling M negatives.
  - Get the updated set of negative candidate $\mathcal{E}'$ by (5).
  - Synthesize a hard negative $\mathbf{e}_{v^-}$ based on $\mathcal{E}'$ by (6).
  - $\mathcal{L} = \mathcal{L} + \ln \sigma(\mathbf{e}_u \cdot \mathbf{e}_{v^-} - \mathbf{e}_u \cdot \mathbf{e}_{v^+})$.
- **end**
- Update θ by descending the gradients $\nabla_\theta \mathcal{L}$.

end

Loss

BPR Loss：

$$\mathcal{L}_{\text{BPR}} = \sum_{\substack{(u,v^+)\in O^+ \\ \mathbf{e}_{v^-}\sim f_{\text{MixGCF}}(u,v^+)}} \ln \sigma(\mathbf{e}_u \cdot \mathbf{e}_{v^-} - \mathbf{e}_u \cdot \mathbf{e}_{v^+}), \tag{9}$$

$$\mathcal{L}_{K-\text{pair}} = \sum_{(u,v^+,v_0^-,\cdots,v_K^-)\in O} -\ln \frac{\exp(y_{u,v^+})}{\exp(y_{u,v^+}) + \sum_{i=0}^{K} \exp(y_{u,v_i^-})}$$

where $\{v_0^-, \cdots, v_K^-\}$ denotes the set of K sampled negatives for each interaction pair.

实验

- 推荐的数据集
  - Yelp
  - Alibaba
  - Amazon

- GNN-based推荐算法
  - LightGCN
  - NGCF
  - PinSage

- Baseline
  - RNS：随机负例抽样策略将均匀分布应用于负例的抽样。
  - DNS：动态负采样策略是目前最先进的采样方法，它自适应地选择推荐得分最高的负例，这个负例即为难负例。
  - IRGAN：IRGAN将推荐集成到一个生成性对抗网络中，在该网络中，生成器充当采样器挑选负样本。
  - AdvIR：AdvIR也是一种对抗性采样器，通过添加对抗性干扰，将对抗性采样与对抗性训练结合起来。
  - MCNS: 提出通过近似正分布对负样本进行采样。

实验

Table 2: Overall Performance Comparison.

	Alibaba		Yelp2018		Amazon	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
LightGCN+RNS	0.0584	0.0275	0.0628	0.0515	0.0398	0.0177
LightGCN+DNS	0.0737	0.0343	0.0695	0.0571	0.0449	0.0211
LightGCN+IRGAN	0.0605	0.0280	0.0641	0.0527	0.0412	0.0185
LightGCN+AdvIR	0.0583	0.0273	0.0624	0.0510	0.0401	0.0185
LightGCN+MCNS	0.0632	0.0284	0.0658	0.0529	0.0423	0.0192
LightGCN+MixGCF	**0.0763***	**0.0357***	**0.0713***	**0.0589***	**0.0460***	**0.0216***
NGCF+RNS	0.0426	0.0197	0.0577	0.0469	0.0294	0.0123
NGCF+DNS	0.0453	0.0207	0.0650	0.0529	0.0312	0.0130
NGCF+IRGAN	0.0435	0.0200	0.0615	0.0502	0.0283	0.0120
NGCF+AdvIR	0.0440	0.0203	0.0614	0.0500	0.0318	0.0134
NGCF+MCNS	0.0430	0.0200	0.0625	0.0501	0.0313	0.0136
NGCF+MixGCF	**0.0544***	**0.0262***	**0.0688***	**0.0566***	**0.0350***	**0.0154***
PinSage+RNS	0.0196	0.0085	0.0410	0.0328	0.0193	0.0080
PinSage+DNS	0.0405	0.0183	0.0590	0.0488	0.0217	0.0088
PinSage+IRGAN	0.0200	0.0090	0.0422	0.0343	0.0248	0.0088
PinSage+AdvIR	0.0196	0.0090	0.0387	0.0313	0.0243	0.0087
PinSage+MCNS	0.0212	0.0095	0.0432	0.0349	0.0202	0.0088
PinSage+MixGCF	**0.0489***	**0.0226***	**0.0632***	**0.0525***	**0.0273***	**0.0124***

DNS在大多数情况下表现仅次于MixGCF

（1）通过hop mixing, MixGCF增强了负例表示，提高了推荐模型的泛化能力。

（2）合成的难负例包含了来自多个实例的不同语义信息，同样为推荐模型提供了一个有效的信息梯度。

消融实验---去除positive mixing

Table 3: Performance of MixGCF without positive mixing.

	Alibaba		Yelp2018		Amazon	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
LightGCN+MixGCF$_{w/o\ p\text{-}m}$	0.0748	0.0347	0.0705	0.0581	0.0448	0.0212
NGCF+MixGCF$_{w/o\ p\text{-}m}$	0.0479	0.0226	0.0674	0.0555	0.0332	0.0147
PinSage+MixGCF$_{w/o\ p\text{-}m}$	0.0486	0.0231	0.0526	0.0432	0.0273	0.0120

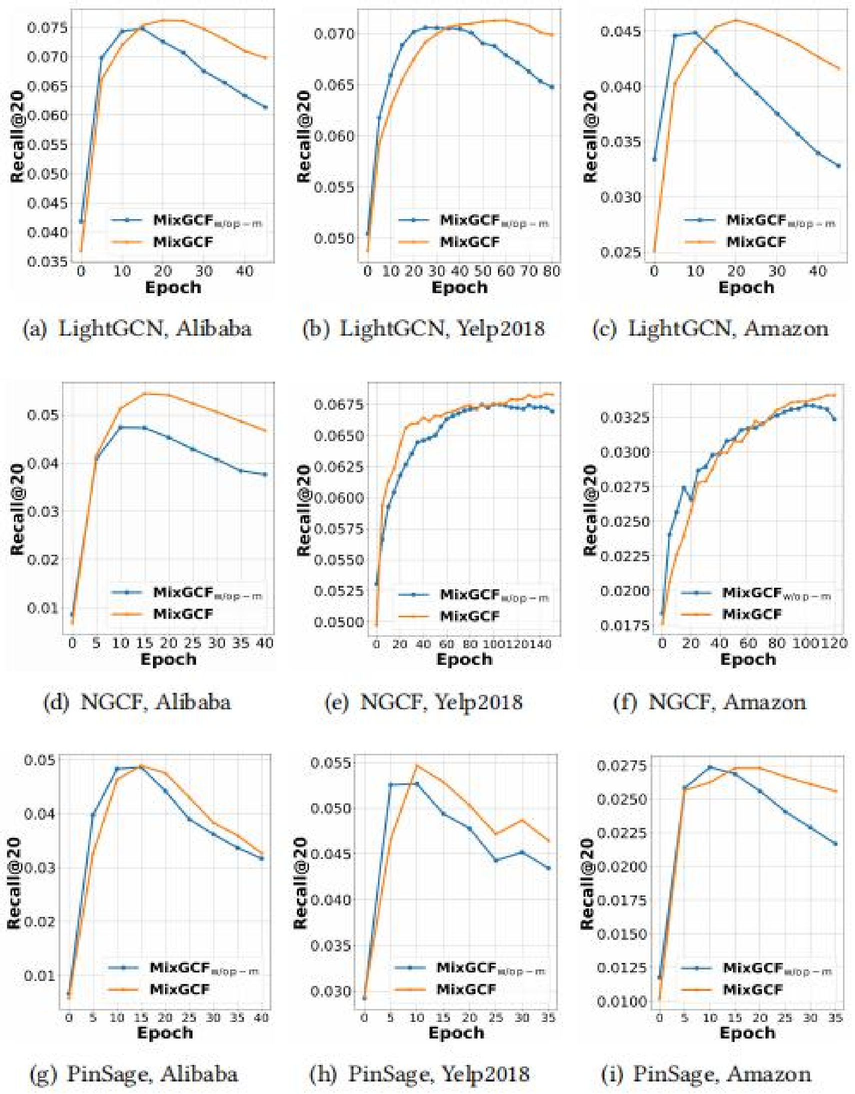


Figure 3: Impact of positive mixing.

消融实验---层数（跳数）L

Table 4: Impact of the number of aggregation modules (L).

	Alibaba		Yelp2018		Amazon	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
LightGCN+MixGCF-1	0.0651	0.0309	0.0684	0.0564	0.0403	0.0193
LightGCN+MixGCF-2	0.0726	0.0335	0.0707	0.0582	0.0438	0.0209
LightGCN+MixGCF-3	0.0763	0.0357	0.0713	0.0589	0.0460	0.0216
NGCF+MixGCF-1	0.0484	0.0234	0.0647	0.0526	0.0320	0.0151
NGCF+MixGCF-2	0.0545	0.0262	0.0664	0.0542	0.0345	0.0153
NGCF+MixGCF-3	0.0544	0.0262	0.0688	0.0566	0.0350	0.0154
PinSage+MixGCF-1	0.0487	0.0231	0.0639	0.0526	0.0289	0.0130
PinSage+MixGCF-2	0.0472	0.0223	0.0627	0.0519	0.0278	0.0121
PinSage+MixGCF-3	0.0489	0.0226	0.0632	0.0525	0.0273	0.0124

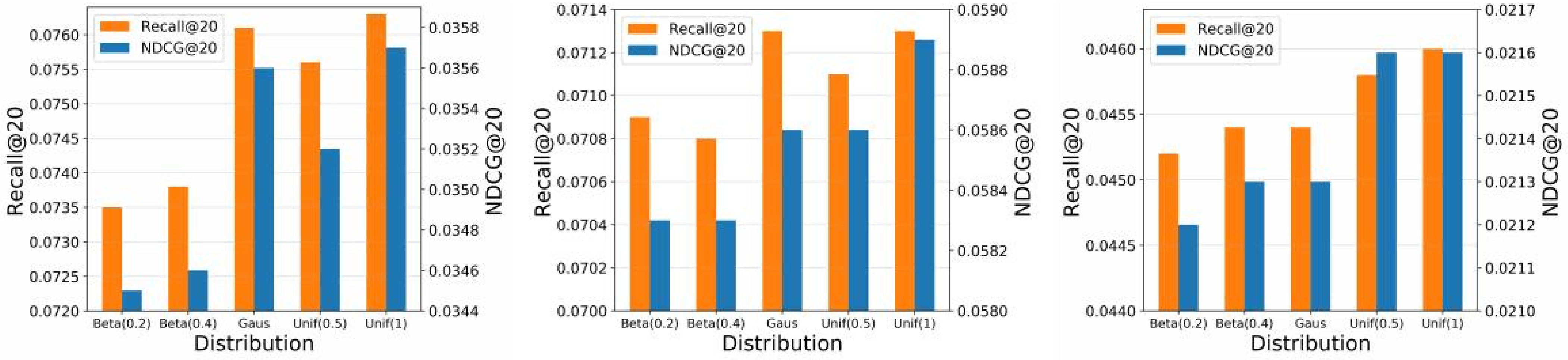
过平滑

消融实验---负例候选集大小M

Table 5: Impact of the size of candidate set (M).

		Alibaba		Yelp2018		Amazon	
		Recall	NDCG	Recall	NDCG	Recall	NDCG
LightGCN+MixGCF	M=8	0.0697	0.0311	0.0664	0.0547	0.0443	0.0203
	M=16	0.0728	0.0339	0.0684	0.0562	**0.0460***	**0.0216***
	M=32	**0.0763***	**0.0357***	0.0703	0.0579	0.0455	0.0215
	M=64	0.0744	0.0355	**0.0713***	**0.0589***	0.0430	0.0206
NGCF+MixGCF	M=8	0.0468	0.0201	0.0627	0.0512	0.0350	0.0147
	M=16	0.0518	0.0237	0.0658	0.0539	0.0333	0.0144
	M=32	0.0532	0.0253	0.0682	0.0560	0.0347	0.0154
	M=64	**0.0544***	**0.0262***	**0.0688***	**0.0566***	**0.0350***	**0.0154***
PinSage+MixGCF	M=8	0.0178	0.0075	0.0495	0.0402	0.0204	0.0072
	M=16	0.0388	0.0173	0.0546	0.0448	0.0207	0.0084
	M=32	0.0435	0.0195	0.0608	0.0501	0.0238	0.0106
	M=64	**0.0489***	**0.0226***	**0.0632***	**0.0525***	**0.0273***	**0.0124***

正样本混合比例α的影响



(a) Alibaba (b) Yelp2018 (c) Amazon

实验

负例个数K的选择

Table 6: Impact of the number of negative instances (K).

		Alibaba		Yelp2018		Amazon	
		Recall	NDCG	Recall	NDCG	Recall	NDCG
LightGCN+MixGCF	K=1	0.0763	0.0357	0.0713	0.0589	0.0460	0.0216
	K=2	0.0763	0.0359	0.0716	0.0591	0.0460	0.0216
	K=4	0.0775	0.0368	0.0722	0.0595	0.0462	0.0217
	K=8	0.0794	0.0378	0.0723	0.0593	0.0464	0.0220
NGCF+MixGCF	K=1	0.0544	0.0262	0.0688	0.0566	0.0350	0.0154
	K=2	0.0582	0.0279	0.0681	0.0560	0.0345	0.0157
	K=4	0.0596	0.0288	0.0688	0.0564	0.0346	0.0156
	K=8	0.0624	0.0305	0.0683	0.0562	0.0345	0.0157
PinSage+MixGCF	K=1	0.0489	0.0226	0.0632	0.0525	0.0273	0.0124
	K=2	0.0529	0.0249	0.0631	0.0523	0.0302	0.0138
	K=4	0.0550	0.0256	0.0640	0.0532	0.0300	0.0137
	K=8	0.0560	0.0261	0.0651	0.0537	0.0300	0.0140

总结

- 提出了一个即插即用的负样本采样方法，不是直接从原始样本中进行负采样，而是通过生成难负样本来进行负采样，以改进基于GNN的推荐系统。
- 展示MixGCF给GNN推荐算法带来的显著改进，以及它在各种负采样技术上的一致表现。

《Structured Graph Convolutional Networks with Stochastic Masks for Recommender Systems》

By：SIGIR 2021
Visa 研究

背景

- GCN中图的稀疏性
  - 稀疏性意味着在消息传递过程中，只有重要的邻居应该连接到目标节点。
- GCN中图的低秩性
  - 低秩约束表明整个图是全局结构的，只有几个因素影响用户的偏好，提升模型泛化性。
- 范数
  - L0 范数： 指向量中非0的元素的个数。如果用L0范数来约束一个参数矩阵W的话，即让参数W是稀疏的。
  - 核范数：指矩阵奇异值的和，如果用核范数来约束一个参数矩阵W的话，那么即求解W秩的凸近似，通常用于约束参数W的低秩性。

动机

- 约束图的稀疏性和低秩性
  - user-item二部图的轻微扰动会导致基于GCN的推荐模型输出错误的预测。
  - 过拟合和过平滑是GCN模型中存在的两大挑战：
    - 过拟合：dropout <——> 不会改变图的邻居矩阵
    - 过平滑：自监督的数据增强 <——> 欠拟合

      (drop node，drop edge)

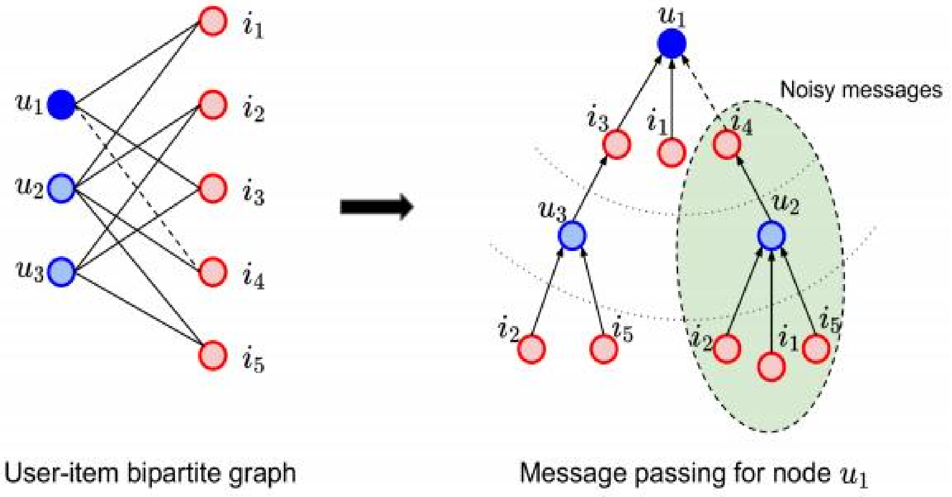


Figure 1: An illustration of how the noisy edge $u_1 - i_4$ will provide misleading information for the target user u_1 under the framework of GCNs.

贡献

- 提出了结构化图卷积神经网络(SGCNs)，通过利用稀疏性和低秩的图结构特性来提高GCNs的性能。
- 为了实现稀疏性，将GCN的每一层附加有一个可训练的随机二元掩膜来去除噪声和不重要的边。为了保证其低秩性质，采用了核范数正则化方法，低秩约束也可以增强gcn的鲁棒性和泛化性。
- 提出了一个随机二元优化的无偏梯度估计器，将随机二元优化转换为可微，以更好地反向传播二元变量的梯度，共同学习随机二元掩膜和GCNs的参数。

Stochastic Binary Masks

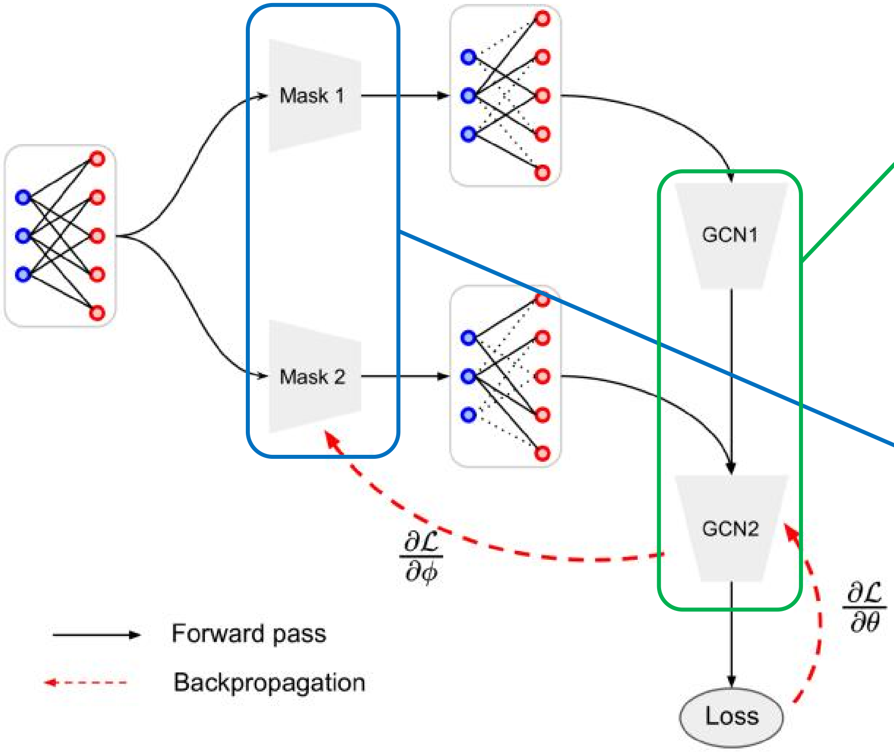


Figure 2: The overview of a two-layer GCN with stochastic binary masks. Here only partial gradients are shown in the backpropagation.

GCN推荐模型

以LightGCN为例

$$\mathbf{E}^{(0)} \in \mathbb{R}^{(|\mathcal{U}|+|\mathcal{I}|)\times d}$$

$$\mathbf{E}^{(k+1)} = \left(\mathbf{D}^{-\frac{1}{2}}\underline{\mathbf{A}}\mathbf{D}^{-\frac{1}{2}}\right)\mathbf{E}^{(k)}, \quad (5)$$

随机二元掩膜 $Z^{(k)}$

通过为每一层生成不固定的图邻接矩阵**A**$^{(k)}$

$$\mathbf{A}^{(k)} = \mathbf{A} \odot \mathbf{Z}^{(k)},$$

$$\mathbf{Z}^{(k)} \in \{0,1\}.$$

$$\boldsymbol{Z}^{(\boldsymbol{k})}_{(\boldsymbol{u},\boldsymbol{v})} = \begin{cases} 1, & user\ u\text{和}item\ i\ \text{的边在第}k\text{层被保留} \\ 0, & \text{否则} \end{cases}$$

$$\mathcal{R}_s = \sum_{k=1}^{K} \left\| \mathbf{Z}^{(k)} \right\|_0 = \sum_{k=1}^{K} \sum_{(u,v)\in\mathcal{G}} \mathbb{I}\left[\mathbf{Z}^{(k)}_{u,v} \neq 0\right],$$

Stochastic Binary Masks

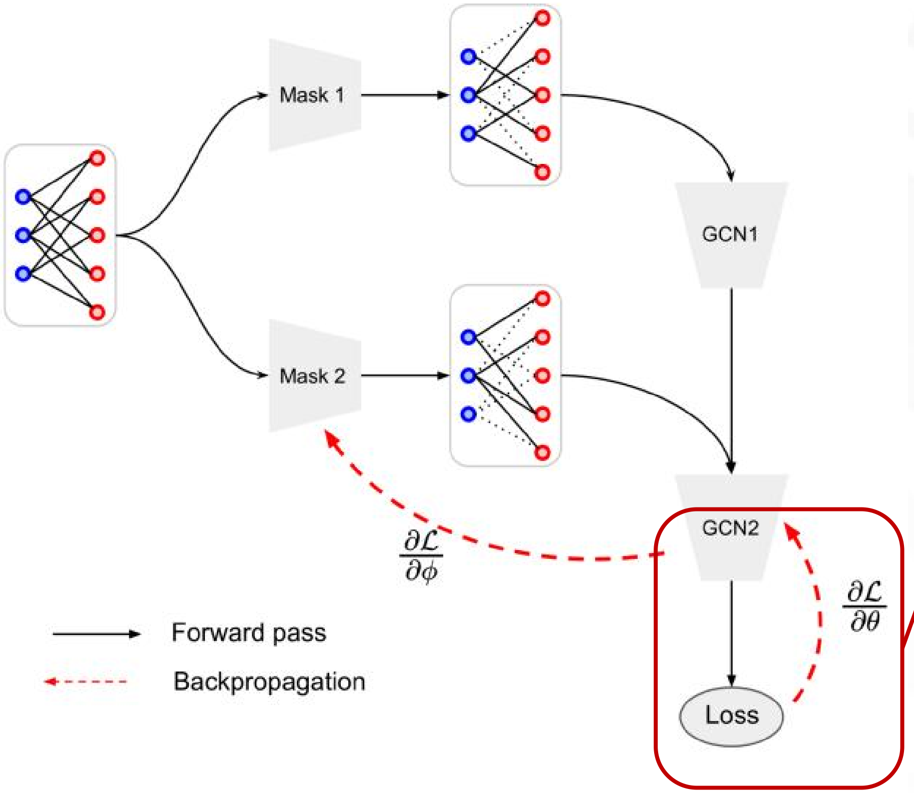


Figure 2: The overview of a two-layer GCN with stochastic binary masks. Here only partial gradients are shown in the backpropagation.

Loss

在GCNs模型Loss中加入稀疏性约束:

$$\mathcal{L}(\mathbf{Z},\Theta) = \mathcal{L}_{BPR}(\{\mathbf{A}\odot\mathbf{Z}^{(k)}\}_{k=1}^{K},\Theta) + \beta\sum_{k=1}^{K}\sum_{(u,v)\in\mathcal{G}}\mathbb{I}\left[\mathbf{Z}_{u,v}^{(k)}\neq 0\right], \quad (9)$$

变分推理: $\mathbf{Z}_{u,v}^{(k)} \sim \text{Bern}(\Pi_{u,v}^{(k)})$.

$E(Z_{u,v}^{(k)}) = \pi_{u,v}^{(k)}$

根据Jensen不等式:

$$L(E_{Z\sim\pi}(Z,\theta)) \le E_{Z\sim\pi}(L(Z,\theta)).$$

因此 Loss 的一个变分上界为:

$$\hat{L} = E_{Z\sim\pi}(L) = E(L_{BPR}(\{A\odot Z^{(k)}\}_{k=1}^{K},\theta) + \beta\sum_{k=1}^{K}\sum_{(u,v)\in G}\boxed{\pi_{u,v}^{(k)}}$$

$$= E_{Z\sim\prod_{k=1}^{K}\text{Bern}(Z^{(k)};\pi^{(k)})}[L_{BPR}(Z,\theta)] + \beta\sum_{k=1}^{K}\sum_{(u,v)\in G}\pi_{u,v}^{(k)}$$

条件独立性假设. ⇕ 每层 $Z^{(k)}$ 相互独立且 $Z^{(k)} \sim \text{Bern}(\pi^{(k)})$.

$Z_{Z^{(0)},Z^{(1)},\dots,Z^{(k)}} \Leftrightarrow \left\{\begin{array}{l} Z^{(0)}\sim\text{Bern}(\pi^{(0)}) \\ \vdots \\ Z^{(k)}\sim\text{Bern}(\pi^{(k)}) \end{array}\right\} \sim \prod_{k=1}^{K}\text{Bern}(z;\pi^{(k)}).$

$$\hat{\mathcal{L}}(\mathbf{Z},\Theta) = \mathbb{E}_{\mathbf{Z}\sim\prod_{k=1}^{K}\text{Bern}(\mathbf{Z}^{(k)};\Pi^{(k)})}\left[\mathcal{L}_{BPR}(\mathbf{Z},\Theta)\right] + \beta\sum_{k=1}^{K}\sum_{(u,v)\in\mathcal{G}}\Pi_{u,v}^{(k)}, \quad (10$$

Stochastic Binary Masks

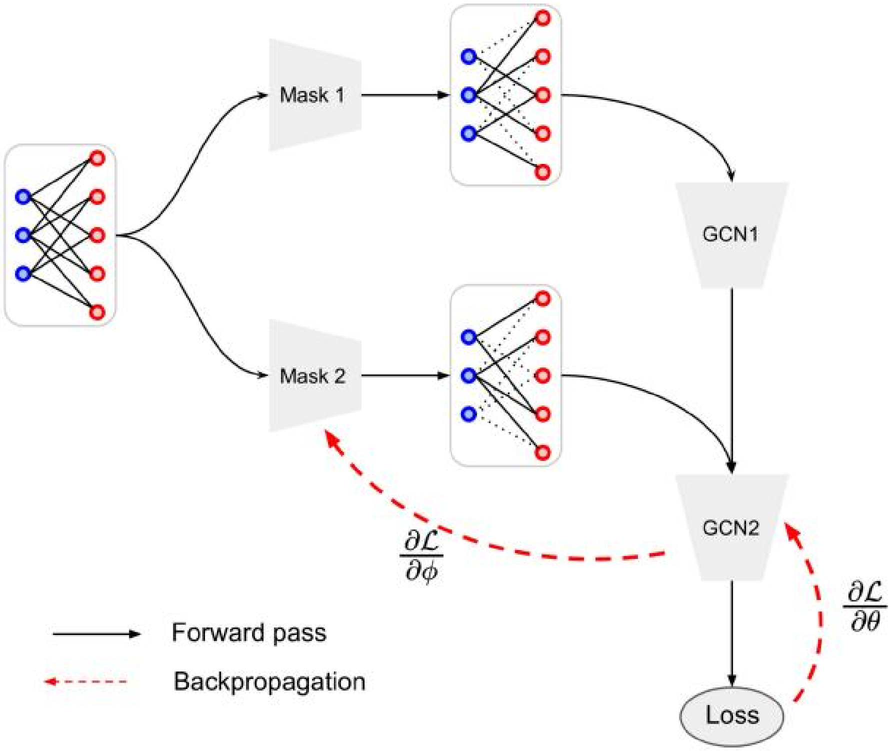


Figure 2: The overview of a two-layer GCN with stochastic binary masks. Here only partial gradients are shown in the backpropagation.

$$\hat{\mathcal{L}}(\mathrm{Z},\Theta) = \mathbb{E}_{\mathrm{Z}\sim\prod_{k=1}^{K}\mathrm{Bern}(\mathrm{Z}^{(k)};\Pi^{(k)})}\left[\mathcal{L}_{BPR}(\mathrm{Z},\Theta)\right] + \beta\sum_{k=1}^{K}\sum_{(u,v)\in G}\Pi_{u,v}^{(k)}, \quad (10)$$

$$\Pi_{u,v}^{(k)} = g(\Phi_{u,v}^{(k)}),$$

sigmoid函数 $\pi = \sigma(\Phi)$

ARM：对于具有一个或多个随机二元层的离散潜变量模型，ARM 估能够直接优化伯努利变量而不引入任何偏差。

对于N位二元随机变量向量 $z = (z_1, \ldots, z_N)^T$ ，

$$\mathcal{E}(\boldsymbol{\phi}) = \mathbb{E}_{z\sim\prod_{i=1}^{N}\mathrm{Bern}(z_i;\sigma(\phi_i))}\left[f(z)\right]$$ 的导数为：

$$\nabla_{\boldsymbol{\phi}}\mathcal{E}(\boldsymbol{\phi}) =$$

$$\mathbb{E}_{\boldsymbol{u}\sim\prod_{i=1}^{N}\mathrm{Uniform}(u_i;0,1)}\left[\left(f(\mathbb{I}[\boldsymbol{u} > \sigma(-\boldsymbol{\phi})]) - f(\mathbb{I}[\boldsymbol{u} < \sigma(\boldsymbol{\phi})])\right)\left(\boldsymbol{u} - \frac{1}{2}\right)\right],$$

where $\mathbb{I}[\boldsymbol{u} > \sigma(-\phi)] := (\mathbb{I}[u_1 > \sigma(-\phi_1)], \ldots, \mathbb{I}[u_N > \sigma(-\phi_N)])^T$, *and* $\sigma(\cdot)$ *is the sigmoid function.*

其中，f 可以代表任意函数，$\boldsymbol{\phi} = (\phi_1, \ldots, \phi_N)^T$ 表示伯努利概率参数的对数 。

Stochastic Binary Masks

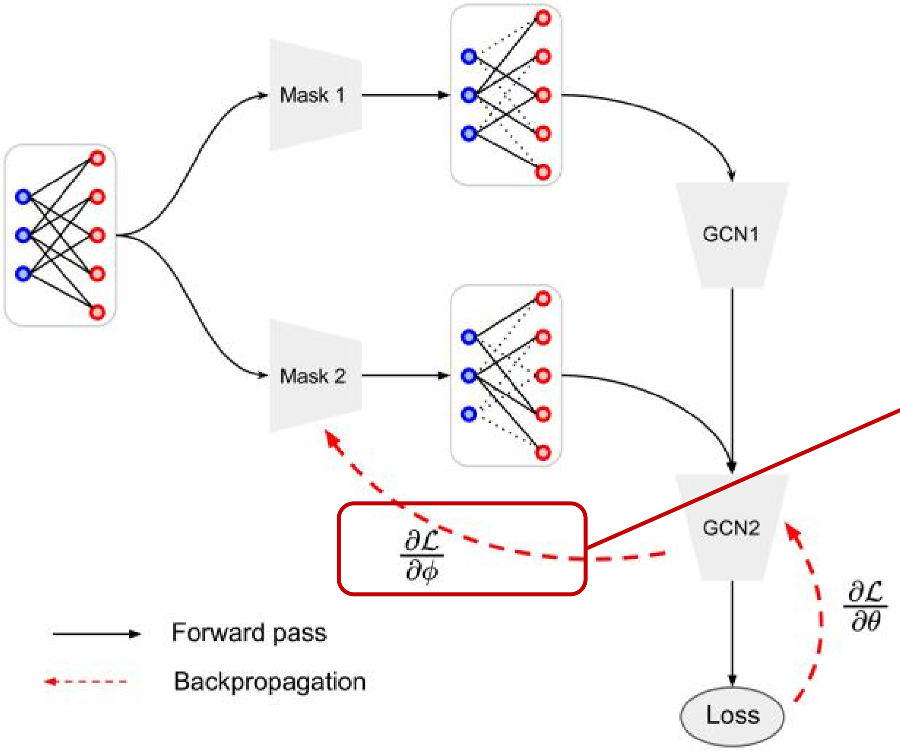


Figure 2: The overview of a two-layer GCN with stochastic binary masks. Here only partial gradients are shown in the backpropagation.

$$\hat{\mathcal{L}}(\mathbf{Z},\Theta) = \mathbb{E}_{\mathbf{Z}\sim\prod_{k=1}^{K}\mathrm{Bern}(\mathbf{Z}^{(k)};\Pi^{(k)})}\left[\mathcal{L}_{BPR}(\mathbf{Z},\Theta)\right] + \beta\sum_{k=1}^{K}\sum_{(u,v)\in\mathcal{G}}\Pi_{u,v}^{(k)}, \tag{10}$$

$$\Pi_{u,v}^{(k)} = g(\Phi_{u,v}^{(k)}),$$ sigmoid函数 $\pi = \sigma(\Phi)$

应用ARM，对$\hat{\mathcal{L}}(\Phi,\theta)$关于参数$\Phi$求导为：

$$\nabla_{\Phi}\hat{\mathcal{L}}(\Phi,\Theta) = \mathbb{E}_{\mathbf{U}\sim\prod_{k=1}^{K}\mathrm{Uniform}(\mathbf{U}^{(k)};0,1)}\left[\left(f(\mathbb{I}[\mathbf{U}>\sigma(-\Phi)]) - f(\mathbb{I}[\mathbf{U}<\sigma(\Phi)])\right)(\mathbf{U}-\frac{1}{2})\right] + \beta\nabla_{\Phi}\sigma(\Phi), \tag{11}$$

inference：

$$\mathbf{A}^{(k)} = \mathbf{A}\odot\mathbf{Z}^{(k)},$$

$$\mathbb{E}(\mathbf{Z}_{u,v}^{(k)}) = \Pi_{u,v}^{(k)} = g(\Phi_{u,v}^{(k)}).$$

$$\mathbb{E}(\mathrm{Z}^{(k)}) = 0 \ if\ g(\Phi_{u,v}^{(k)}) \le 0.5$$

Low-rank Approximation

$$\mathcal{R}_l = \sum_{k=1}^{K} \left\| \mathbf{A}^{(k)} \right\|_* = \sum_{k=1}^{K} \sum_i \lambda_i (\mathbf{A}^{(k)}), \quad (12)$$

其中，$\lambda_i(\mathrm{A}^{(k)})$表示$\mathrm{A}^{(k)}$的第$i$大奇异值。

SVD的反向传播公式[*]为：

$$\mathbf{K}_{i,j} = \begin{cases} \frac{1}{\lambda_i^2 - \lambda_j^2}, & i \neq j \\ 0, & i = j. \end{cases}$$

PI (Power Iteration)：对初始化奇异向量很敏感。

Proposition 1 (SVD Variations) *Let $X = U\Sigma V^\top$ with $X \in \mathbb{R}^{m,n}$ and $m > n$, such that $U^\top U = I$, $V^\top V = I$ and Σ possessing diagonal structure. Then*

$$d\Sigma = (U^\top dXV)_{diag} \quad (8)$$

and

$$dV = 2V\left(K^\top \circ (\Sigma^\top U^\top dXV)_{sym}\right) \quad (9)$$

with

$$K_{ij} = \begin{cases} \frac{1}{\sigma_i^2 - \sigma_j^2}, & i \neq j \\ 0, & i = j \end{cases} \quad (10)$$

Consequently the partial derivatives are

$$\frac{\partial L}{\partial X} = U\left\{2\Sigma\left(K^\top \circ \left(V^\top \frac{\partial L}{\partial V}\right)\right)_{sym} + \left(\frac{\partial L}{\partial \Sigma}\right)_{diag}\right\}V^\top \quad (11)$$

* Catalin Ionescu, Orestis Vantzos, and Cristian Sminchisescu. 2015. Matrix backpropagation for deep networks with structured layers. In CVPR. 2965–2973.

SGCNs

Low-rank Approximation

$$\mathcal{R}_l = \sum_{k=1}^{K} \left\|\mathbf{A}^{(k)}\right\|_* = \sum_{k=1}^{K} \sum_{i} \lambda_i(\mathbf{A}^{(k)}), \quad (12)$$

top-n 截断SVD

$$\mathcal{R}_l \approx \sum_{k=1}^{K} \sum_{i}^{n} \lambda_i(\mathbf{A}^{(k)})$$

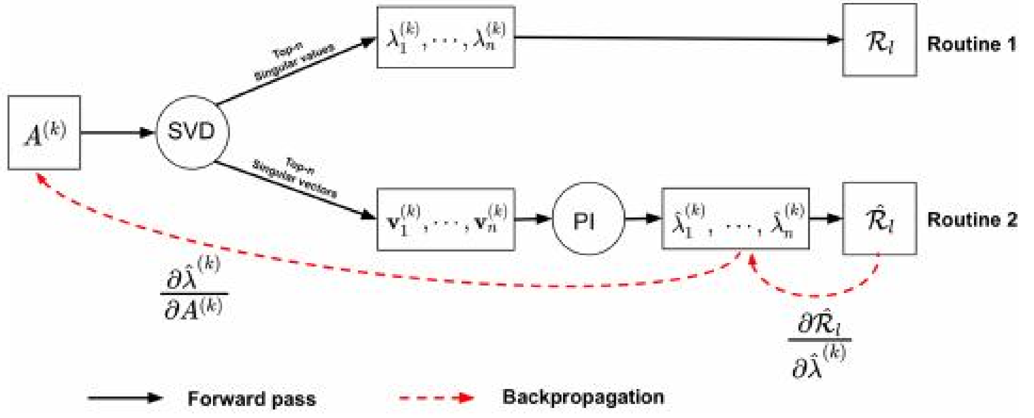


Figure 3: Computational graph for the nuclear norm loss $\mathcal{R}_l$ in Eq. (12). The shadow loss $\hat{\mathcal{R}}_l$ is only used in backpropagation.

正向传播：使用截断的SVD分解每层图邻接矩阵A $^{(k)}$，并得到奇异向量和奇异值，基于该奇异值计算核范数。
反向传播：直接将SVD得到的奇异向量作为PI的初始化进而基于PI进行反向传播，更新A $^{(k)}$。

Wei Wang, Zheng Dang, Yinlin Hu, Pascal Fua, and Mathieu Salzmann. 2019. Backpropagation-friendly eigendecomposition. In NeurIPS. 3162–3170.

联合训练

Algorithm 1: SGCN

Input: The training graph $\mathbf{A}$, the number of GCN layers K, and the regularization coefficients α, β, and γ.

```
for each mini-batch do
    for k ← 1 to K do
        Generate a subgraph A^(k) via the stochastic binary mask
          in Eq. (7);
        Feed A^(k) into the k-th layer of GCN;
    end
    Compute the loss L_SGCN in Eq. (13);
    Update the parameters of GCN and stochastic binary masks;
end
```

Output: A well-trained SGCN to predict $\hat{y}_{ui}$.

$$\mathcal{L}_{SGCN} = \mathcal{L}_{BPR} + \beta \cdot \mathcal{R}_s + \gamma \cdot \mathcal{R}_l \quad (13)$$

实验

- 推荐的数据集
  - Movielens-1M
  - Gowalla
  - Yelp
  - Amazon

- Baseline
  - CF-based methods
    - BPR-MF
    - NeuMF
    - GC-MC
  - GCN-based methods
    - NGCF
    - LightGCN
    - SGCNs（S-NGCF，S-LightGCN）

实验

Table 2: Recommendation performance comparison for different models. Note that R and N are short for Recall and NDCG, respectively. %Improv denotes the relative improvement of SGCNs over their corresponding GCNs. The best results are highlighted in bold and the second best ones are underlined.

	MovieLens				Gowalla				Yelp				Amazon			
Metric	R@50	N@50	R@100	N@100	R@50	N@50	R@100	N@100	R@50	N@50	R@100	N@100	R@50	N@50	R@100	N@100
BPR-MF	0.282	0.243	0.371	0.354	0.129	0.118	0.346	0.156	0.093	0.038	0.140	0.047	0.069	0.041	0.122	0.059
NeuMF	0.297	0.251	0.378	0.368	0.143	0.124	0.350	0.169	0.103	0.040	0.151	0.050	0.074	0.047	0.135	0.061
GC-MC	0.291	0.247	0.375	0.360	0.137	0.122	0.347	0.163	0.098	0.036	0.146	0.044	0.070	0.044	0.128	0.064
HOP-Rec	0.314	0.260	0.373	0.367	0.135	0.125	0.352	0.182	0.111	0.048	0.163	0.053	0.080	0.059	0.143	0.074
BiNE	0.312	0.253	0.381	0.371	0.141	0.126	0.354	0.188	0.110	0.042	0.155	0.049	0.076	0.052	0.134	0.069
NGCF	0.325	0.289	0.393	0.382	0.160	0.132	0.356	0.197	0.114	0.054	0.172	0.061	0.092	0.065	0.157	0.076
S-NGCF	<u>0.341</u>	<u>0.311</u>	<u>0.417</u>	**0.408**	<u>0.177</u>	<u>0.156</u>	<u>0.384</u>	<u>0.218</u>	<u>0.127</u>	<u>0.068</u>	<u>0.194</u>	<u>0.077</u>	<u>0.107</u>	<u>0.074</u>	<u>0.170</u>	<u>0.087</u>
%Improv.	4.92%	7.61%	6.11%	6.81%	10.63%	18.18%	7.87%	10.66%	11.40%	25.93%	12.79%	26.23%	16.30%	13.85%	8.28%	14.47%
LightGCN	0.328	0.294	0.399	0.384	0.163	0.134	0.360	0.205	0.117	0.059	0.181	0.067	0.098	0.071	0.162	0.083
S-LightGCN	**0.347**	**0.313**	**0.424**	<u>0.406</u>	**0.178**	**0.159**	**0.387**	**0.223**	**0.134**	**0.073**	**0.199**	**0.081**	**0.114**	**0.078**	**0.177**	**0.092**
%Improv.	5.79%	6.46%	6.27%	5.73%	9.20%	18.66%	7.50%	8.78%	14.53%	23.73%	9.94%	20.90%	16.33%	9.86%	9.56%	10.84%

实验

鲁棒性分析

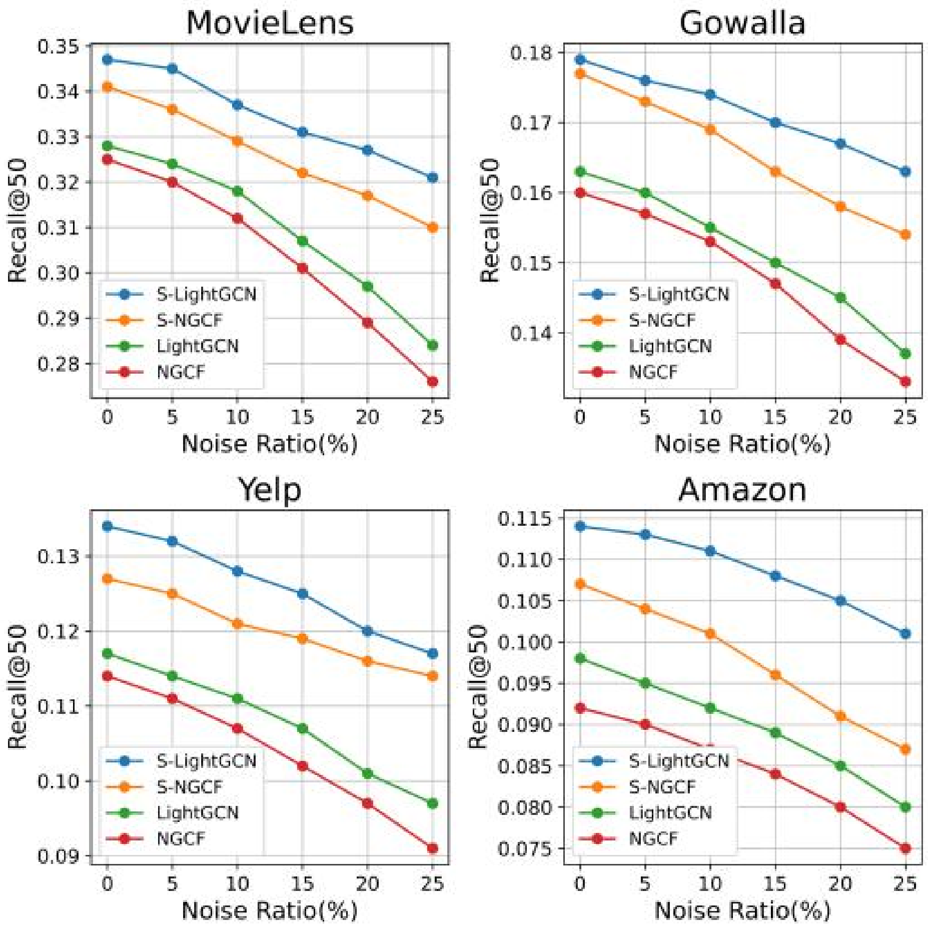


Figure 4: The model robustness of SGCNs and GCNs to the different levels of noisy edges.

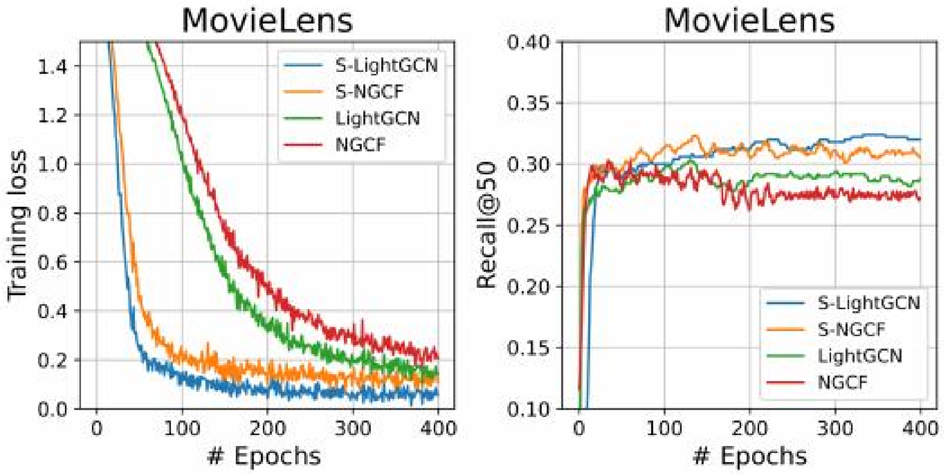


Figure 5: Analysis of over-fitting for SGCNs and GCNs, which are evaluated by training loss and testing recall for noisy MovieLens dataset.

超参实验

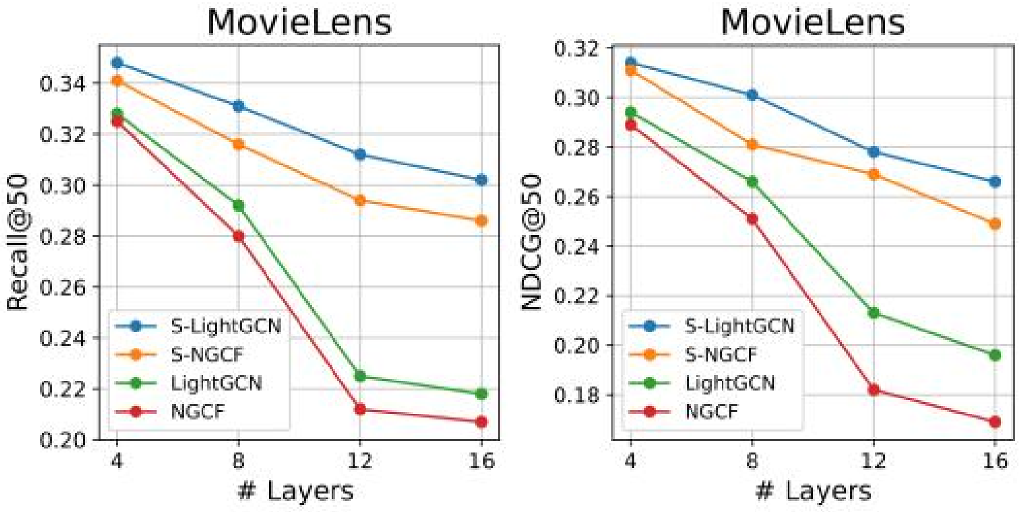


Figure 6: Analysis of over-smoothing for SGCNs and GCNs.

过平滑

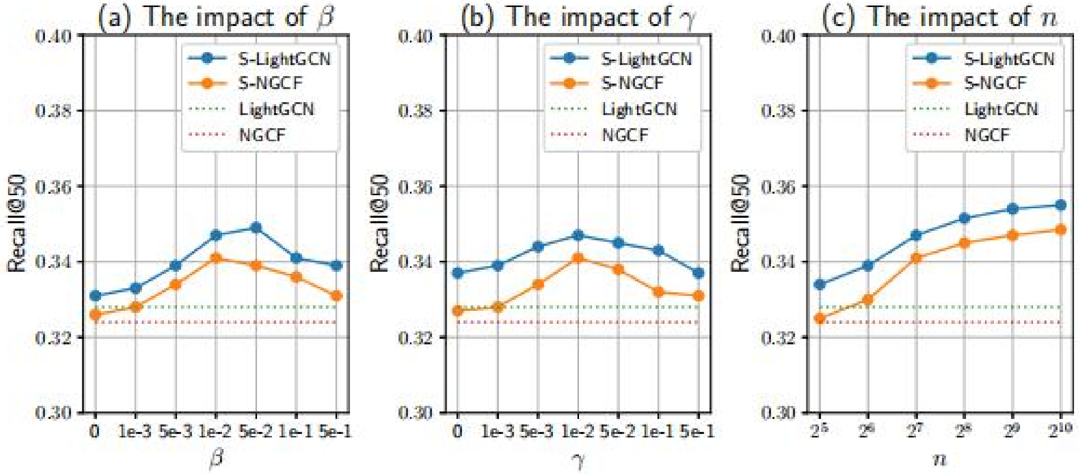


Figure 7: Results of parameter sensitivity on MovieLens.

总结

- 稀疏性：使用一个可训练的随机二元掩膜来去除噪声和不重要的边，为GCN的每一层生成一个稀疏的图邻接矩阵，并应用了一个随机二元优化的无偏梯度估计器反向传播二元变量的梯度，共同学习随机二元掩膜和GCNs的参数。
- 低秩性：采用核范数正则化方法，采用了截断SVD和PI相结合的方式更新稀疏图邻接矩阵。
- 可解释性：为目标用户发现一个紧凑的子图结构

Thanks！

Q & A