DGR: A General Graph Desmoothing Framework for Recommendation via Global and Local Perspectives

Leilei Ding[1] , **Dazhong Shen**[2] , **Chao Wang**[3,1*] , **Tianfu Wang**[1] , **Le Zhang**[4] , **Yanyong Zhang**[1*]

[1]School of Computer Science and Technology, University of Science and Technology of China
[2]Shanghai Artificial Intelligence Laboratory
[3]Guangzhou HKUST Fok Ying Tung Research Institute
[4]Business Intelligence Lab, Baidu Research
{dingleilei,tianfuwang}@mail.ustc.edu.cn, dazh.shen@gmail.com, chadwang2012@gmail.com, zhangle0202@gmail.com, yanyongz@ustc.edu.cn

Abstract

Graph Convolutional Networks (GCNs) have become pivotal in recommendation systems for learning user and item embeddings by leveraging the user-item interaction graph's node information and topology. However, these models often face the famous over-smoothing issue, leading to indistinct user and item embeddings and reduced personalization. Traditional desmoothing methods in GCN-based systems are model-specific, lacking a universal solution. This paper introduces a novel, model-agnostic approach named **D**esmoothing Framework for **GCN**-based **R**ecommendation Systems (**DGR**). It effectively addresses over-smoothing on general GCN-based recommendation models by considering both global and local perspectives. Specifically, we first introduce vector perturbations during each message passing layer to penalize the tendency of node embeddings approximating overly to be similar with the guidance of the global topological structure. Meanwhile, we further develop a tailored-design loss term for the readout embeddings to preserve the local collaborative relations between users and their neighboring items. In particular, items that exhibit a high correlation with neighboring items are also incorporated to enhance the local topological information. To validate our approach, we conduct extensive experiments on 5 benchmark datasets based on 5 well-known GCN-based recommendation models, demonstrating the effectiveness and generalization of our proposed framework. Our code is available at https://github.com/me-sonandme/DGR.

1 Introduction

Personalized recommendation is a vital technology in various domains, such as social media, e-commerce, and career recommendation, aiming to recommend items based on users' interests by mining and matching the embeddings of users

and candidate items [Wang *et al.*, 2021b; Wang *et al.*, 2021a; Li *et al.*, 2024]. The interaction information between users and items can naturally be represented as a bipartite graph, making Graph Convolutional Networks (GCNs) [He *et al.*, 2020] an effective approach to learning user and item embedding vectors. Specifically, conventional GCN-based recommendation models, such as NGCF [Wang *et al.*, 2019a], LightGCN [He *et al.*, 2020] and DGCF [Wang *et al.*, 2020b] exclusively leverage the message passing mechanism to aggregate local information and directly incorporate the signals captured from historical interaction data into the embedding vectors, which have witnessed both the training efficiency and better performance.

Unfortunately, as mentioned in [Chen *et al.*, 2020b; Liu *et al.*, 2021; Chen *et al.*, 2022a; Cai *et al.*, 2023; Zhou *et al.*, 2023; Peng *et al.*, 2022; Wang *et al.*, 2023b; Xia *et al.*, 2023], GCN-based recommendation models face the inherent over-smoothing problem, where all user and item embeddings converge toward similarity during the message passing process. It causes a failure in personalized recommendation, resulting in the same recommendation results for every user [Zhou *et al.*, 2023]. Moreover, data in recommendation systems are often highly sparse, making it more difficult to model relationships between users and items due to limited direct user-item interaction information. Consequently, the existing GCN-based recommendation struggles to accurately capture the differences in the fine-grained relationships among users and items.

Recently, significant efforts have been devoted to addressing the over-smoothing problem in GCN-based recommendation. For example, many well-known GCN-based recommendation models, such as LightGCN [He *et al.*, 2020] and NGCF [Wang *et al.*, 2019a], opt to incorporate outputted embeddings from each layer into the final readout embeddings, preserving node information characteristics and slightly alleviating the over-smoothing problem. Meanwhile, there are also some works proposed to adjust the message passing processes [He *et al.*, 2023; Peng *et al.*, 2022; Liu *et al.*, 2021] or adopt self-supervised learning to maintain the distinctiveness of user and item embeddings [Xia *et al.*, 2023; Xia *et al.*, 2022; Cai *et al.*, 2023; Wu *et al.*, 2021; Lin *et al.*, 2022]. However, most of these approaches are

---

*Chao Wang and Yanyong Zhang are both corresponding authors

specifically designed for individual models, leaving a strong need for a general desmoothing framework in GCN-based recommendation systems.

Indeed, to make full use of the sparse information from the user-item interactions, it is essential to take into consideration both the global and local topological perspectives with model-free solutions. Specifically, on the one hand, with the mathematical nature of the global topological structure, the message passing mechanisms in existing GCN models often lead to the rapid convergence of all node embeddings and cause over-something. Therefore, the first challenge lies in modifying the message passing layer to distinguish user item embeddings. On the other hand, local collaborative relations, which encourage similar embeddings for the neighbors, are also critical for recommendation performance [Wang *et al.*, 2023a; Mao *et al.*, 2021]. It is also challenging to mine those collaborative signals from the local topological structure during desmoothing process.

In this paper, we introduce the **D**esmoothing Framework for **G**CN-based **R**ecommendation Systems (**DGR**), a novel, universally applicable, and easy-to-use approach to tackle the over-smoothing problem from both global and local perspectives and enhance personalized recommendation in GCN-based recommendation systems. Our approach encompasses two primary modules: **G**lobal Desmoothing **M**essage **P**assing (**GMP**) and **L**ocal Node **E**mbedding **C**orrection (**LEC**). The GMP module is designed to counteract the global trend of user and item embeddings approximating overly to the limit state of infinite-layer graph convolutions, i.e., the over-smoothing point. It introduces vector perturbations at each layer to steer embeddings away from the over-smoothing point. Simultaneously, the LEC module focuses on preserving and emphasizing the local collaborative signals, particularly in sparse data scenarios. It works by maintaining local collaborative relationships between users and neighboring items and incorporating highly correlated items to strengthen the local topological information. Differing from previous methods that targeted improvements in individual GCN-based recommendation systems, our DGR framework is a general, model-free solution, easily adaptable across various GCN-based models. To substantiate our claims, we have rigorously tested DGR with five popular recommendation models across five public benchmark datasets. The results from these extensive experiments, along with several case studies, have convincingly demonstrated the effectiveness and broad applicability of our framework in enhancing personalized recommendations in GCN-based systems.

2 Preliminaries

In this section, we first introduce the essential background knowledge about GCN-based recommendation systems and the concept of the over-smoothing point.

2.1 GCNs in Recommendation Systems

GCNs are the representative models for complex graph-structured data [Wang *et al.*, 2023c; Chen *et al.*, 2022b; Chen *et al.*, 2022c] and have obtained great success in recommendation, where the interaction information between

users and items can naturally be represented as a bipartite graph [Wu *et al.*, 2024]. LightGCN is one of the foundational frameworks for many GCN-based recommendation models [Cai *et al.*, 2023; Yu *et al.*, 2023; Shuai *et al.*, 2022]. It simplifies GCNs by eliminating non-linear feature transformations, retaining only the message passing. Each layer is defined as:

$$\mathbf{E}^{(k+1)} = \widetilde{\mathbf{A}}\mathbf{E}^{(k)} = \widetilde{\mathbf{A}}^{(k+1)}\mathbf{E}^{(0)}, \tag{1}$$

where $\mathbf{E}^{(k)} \in \mathbb{R}^{(N_u+N_i) \times T}$ is the k-th layer node embedding matrix, T is the embedding size, N_u and N_i denote the number of users and items, $\widetilde{\mathbf{A}} \in \mathbb{R}^{(N_u+N_i) \times (N_u+N_i)}$ is the normalization of the adjacent matrix $\mathbf{A}$ of the user-item graph with self-loop, i.e.,

$$\widetilde{\mathbf{A}} = \hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{A}\hat{\mathbf{D}}^{-\frac{1}{2}}, \quad \mathbf{A} = \begin{pmatrix} \mathbf{I}_{N_u} & \mathbf{R} \\ \mathbf{R}^T & \mathbf{I}_{N_i} \end{pmatrix}, \tag{2}$$

where $\mathbf{R} \in \mathbb{R}^{N_u \times N_i}$ is the user-item interaction matrix, each entry $\mathbf{R}_{ui}$ is 1 if user u has interacted with item i otherwise 0. $\mathbf{I}_{N_u}$ and $\mathbf{I}_{N_i}$ are identity matrics with sizes of N_u and N_i, respectively. $\hat{\mathbf{D}}$ is a $(N_u+N_i) \times (N_u+N_i)$ diagonal matrix, in which each entry $\hat{\mathbf{D}}_{ij}$ denotes the number of nonzero entries in the i-th row vector of the adjacency matrix $\mathbf{A}$.

Note that, the initial definition of the LightGCN layer has removed the self-loop connections on nodes. However, as the original paper claims, the layer combination operation plays a similar role by summing the weighted node embeddings at each layer as the final output representation. Along this line, we can rewrite the above message passing process for user embedding $\mathbf{e}_u$ and item embedding $\mathbf{e}_i$ as follows,

$$\begin{aligned} \mathbf{e}_u^{(k+1)} &= \frac{\mathbf{e}_u^{(k)}}{d_u+1} + \sum_{i \in V_u} \frac{\mathbf{e}_i^{(k)}}{\sqrt{(d_u+1)(d_i+1)}}, \\ \mathbf{e}_i^{(k+1)} &= \frac{\mathbf{e}_i^{(k)}}{d_i+1} + \sum_{i \in V_i} \frac{\mathbf{e}_u^{(k)}}{\sqrt{(d_i+1)(d_u+1)}}, \end{aligned} \tag{3}$$

where V_u and V_i denote the neighbor node set of user u and item i with size of d_u and d_i, respectively. The readout node embeddings $\mathbf{e}_u$ and $\mathbf{e}_i$ are the combination of embeddings outputted by each message passing layer. Then, the model prediction is defined as the inner product of user and item final embedding representations:

$$\hat{y}_{u,i} = \mathbf{e}_u^\mathsf{T}\mathbf{e}_i, \tag{4}$$

which is used as the ranking score for recommendation. Meanwhile, the Bayesian Personalized Ranking (BPR) loss is employed, which is a pairwise loss that encourages the possibility of an observed interaction to be higher than its unobserved counterparts, i.e.,

$$\mathcal{L}_{\text{BPR}} = - \sum_{(u,i,j) \in \mathcal{D}} \log \sigma(\hat{y}_{ui} - \hat{y}_{uj}), \tag{5}$$

where $\sigma(\cdot)$ is the sigmoid function, $\mathcal{D} = \{(u,i,j) \mid R_{u,i} = 1, R_{u,j} = 0\}$ denotes to the pairwise training dataset, and j denotes the sampled item that user u has not interacted with.
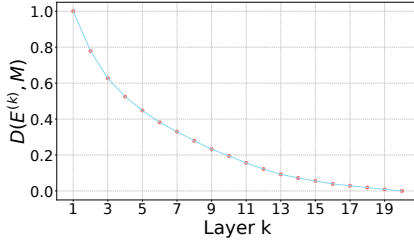
Figure 1: Empirical analysis on the evolution of the embeddings as the layer number of message-passing increases. It shows the $\mathbf{D}(\mathbf{E}^{(k)}, \mathbf{M})$ changes across the layer number k in the range [1, 20].

2.2 Over-Smoothing Point

However, the GCN-based recommendation usually suffers performance degradation as the number of message passing layers k increases due to the well-known over-smoothing problem [He *et al.*, 2020; Wang *et al.*, 2019b]. Specifically, the node embedding $\mathbf{e}^k$ (the row of $\mathbf{E}^{(k)}$) will all converge to the corresponding final point when k tends to infinity, called the **over-smoothing point**. Theoretically according to Theorem 1 in [Chen *et al.*, 2020c], we can derive that:

$$\widetilde{\mathbf{A}}_{u,i}^{\infty} = \lim_{k\to\infty} \widetilde{\mathbf{A}}_{i,j}^{k} = \frac{\sqrt{(d_u + 1)(d_i + 1)}}{2N_{edge} + N_i + N_u}, \qquad (6)$$

where d_u, d_i represent the degrees of user node u and item node i, respectively. N_{edge} is the total number of edges in the user-item graph. We can then obtain Over-smoothing Steady State Matrix $\mathbf{M}$ via:

$$\mathbf{M} = \lim_{k\to\infty} \mathbf{E}^{(k)} = \widetilde{\mathbf{A}}^{\infty} \mathbf{E}^{(0)}, \qquad (7)$$

where we denote the **over-smoothing point** as $\mathbf{m}$, which is the row of $\mathbf{M}$. According to [Chen *et al.*, 2020c], the rows of $\mathbf{M}$ are linearly correlated with each other. In other words, the high-order power of normalized adjacent matrix $\widetilde{\mathbf{A}}$ tends to undergo a rank collapse as k increases.

Actually, in the context of recommendation, the over-smoothing problem can be more serious due to the long tail and the high sparsity. On the one hand, in the item-user graph, most interactions occur on the popular users or items, which have a higher degree than most common nodes. As a result, with Equation 7, those nodes' embeddings would converge at a higher rate and have a stronger impact on pushing the convergence of other nodes' embeddings. On the other hand, the high sparsity makes nodes interact more with local neighborhoods, due to limited direct user-item interactions. As a result, the node embeddings prefer to converge more rapidly.

To provide visual insights of the evolution of the node embeddings as the layer number of message-passing increases, we conduct an empirical analysis on MovieLens1M dataset in Figure 1 to illustrate the evolution of user or item embeddings during the message passing process. Specifically, we calculate the distance between all node embeddings and over-smoothing steady state matrix as follows:

$$\mathbf{D}\left(\mathbf{E}^{(k)}, \mathbf{M}\right) = \frac{1}{n} \sum_{i \in [n]} \left\| \mathbf{e}_i^{(k)} - \mathbf{m}_i \right\|_2, \qquad (8)$$
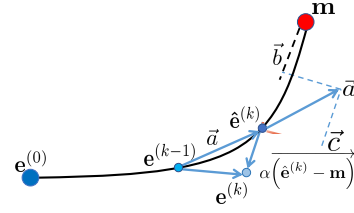

Figure 2: Illustration of the embedding vector updating in Global Desmoothing Message Passing. The solid black line from $\mathbf{e}^{(0)}$ to $\mathbf{m}$ represents the original trajectory of the GCN.

where $\mathbf{D}\left(\mathbf{E}^{(k)}, \mathbf{M}\right)$ is the mean Euclidean distance between the k-th layer node embeddings $\mathbf{E}^{(k)}$ and over-smoothing steady state matrix $\mathbf{M}$. The $\mathbf{e}_i^{(k)}$ and $\mathbf{m}_i$ are the i-th row of $\mathbf{E}^{(k)}$ and $\mathbf{M}$ respectively. Figure 1 shows a rapidly decreasing trend as the layer k increases, indicating that node embeddings tend to converge towards the over-smoothing point as the layer number of message-passing increases. This convergence phenomenon poses a risk of recommendation failure.

3 Our Framework

In this section, we present the technical details of our proposed **D**esmoothing Framework for **G**CN-based **R**ecommendation Systems (**DGR**). As shown in Figure 3, our DGR model is a model-free approach and can be exploited for enhancing various GCN-based recommendation systems by plugging into two components: **G**lobal Desmoothing **M**essage **P**assing (**GMP**) and **L**ocal Node **E**mbedding **C**orrection (**LEC**), which prevents global user and item embeddings from over-smoothing point but maintain similarities among their neighbor nodes. In the following, we will introduce those two components respectively, with a necessary analysis of the complexity.

3.1 Global Desmoothing Message Passing

As discussed in Section 2.2, due to the mathematical nature of global topological structure, the embedding $\mathbf{e}^{(k)} \in \mathbf{E}^{(k)}$ of each user or item will converge to the corresponding over-smoothing point $\mathbf{m} \in \mathbf{M}$ as stacking GCN layers. As a result, each user or item embedding is linearly correlated with each other and cannot be distinguished. To prevent the over-smoothing problem, an effective way is to slow down the convergence rate during each message passing step. Along this line, we propose a simple but effective plug-in module after each message passing layer to perturb the node embeddings away from the final over-smoothing point. Figure 2 provides an illustrative understanding of our proposed approach.

Specifically, the black curve represents the updating trajectory of an embedding vector for a user or an item, approximating the over-smoothing point (the red point). After the k-th original message passing steps, the node embedding evolves from $e^{(k-1)}$ to $\hat{e}^{(k)}$ as Equation 3. We denote the updating vector as $\vec{a} = \hat{e}^{(k)} - e^{(k-1)}$. It can be decomposed into two orthogonal components $\vec{b}$ and $\vec{c}$, where $\vec{b}$ is parallel to the direction of $e^{k-1} - m$ and $\vec{c}$ is perpendicular to it. Intuitively, $\vec{b}$ pushes the embedding vector to the corresponding
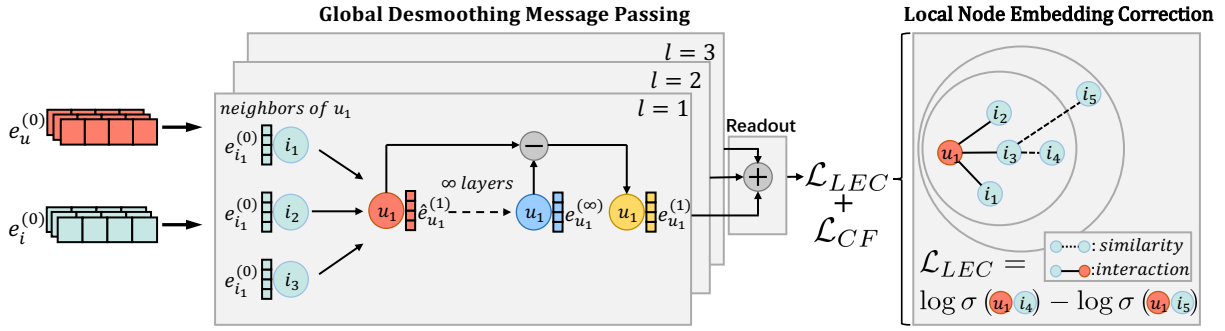
Figure 3: An illustration of DGR architecture. In the Global Desmoothing Message Passing component, user and item embeddings are put away from the over-smoothing point to preserve node embedding distinctiveness. In the Local Node Embedding Correction component, similar neighbor nodes like node i_4 are aggregated together while the marginal nodes like node i_5 is segregated in the local graph to utilize the collaborative signals. In the readout phase, $\mathbf{e}^{(0)}$ are also utilized, although they are not shown in the diagram for clarity.

over-smoothing point. Therefore, we punish the vector updating in this direction with a parameter α:

$$
\begin{aligned}
\mathbf{e}^{(k)} &= \hat{\mathbf{e}}^{(k)} - \alpha(\mathbf{m} - \hat{\mathbf{e}}^{(k)}) = (1 + \alpha)\hat{\mathbf{e}}^{(k)} - \alpha\mathbf{m}, \\
\hat{\mathbf{e}}^{(k)} &= AGG\left(\mathbf{e}_j^{(k-1)}, \{j \in \mathcal{N}\}\right),
\end{aligned}
\tag{9}
$$

where $\mathcal{N}$ denotes the set of neighbor items or users, and AGG denotes the aggregation operation defined in Equation 3. Note that $\vec{b}$ and $\mathbf{m} - \hat{\mathbf{e}}^{(k)}$ have the same direction (without sign) but different scales. Here, we use the latter to perturb the node embedding, abstaining from the use of matrix orthogonal decomposition in the computation, which enhances both robustness and usability.

Our approach to desmooth during message passing is characterized by its simplicity and intuitive nature. In practice, it has demonstrated remarkable effectiveness. Actually, we can also validate the effectiveness with the following proposition.

Proposition 1. *With the plug-in module defined in Equation 9, the distance between any node embedding $\mathbf{e}$ and the corresponding over-smoothing point $\mathbf{m}$ would increase, i.e,*

$$
||\mathbf{e}^{(k)} - \mathbf{m}||_p > ||\hat{\mathbf{e}}^{(k)} - \mathbf{m}||_p,
\tag{10}
$$

where $|| \cdot ||_p$ is the p-norm distance between vectors.

The proof is obvious by bringing Equation 9 into Equation 10. The above inequality suggests that our approach prevents the collapse of user or item embeddings into the over-smoothing point by increasing the distance between them.

3.2 Local Node Embedding Correction

In this subsection, we further focus on how to deal with the over-smoothing problem from the local graph perspective. Differently from the global view, the similarities between some local neighbor nodes are beneficial for collaborative filtering. It is challenging to maintain collaborative relations from local user-item interactions while maintaining distinctiveness among all node embeddings.

Here, we propose a general additional loss for GCN-based recommendations to handle the local user and item embedding learning process. Intuitively, the main idea is to preserve the similarity of the readout embeddings of highly neighbor

and correlated item and user nodes. Meanwhile, The borders between similar nodes and unrelated nodes are referred to as the boundary nodes. Previous studies [Abbasnejad *et al.*, 2020; Goyal *et al.*, 2019; Wang *et al.*, 2021c] have shown that samples in boundary nodes are often discriminative in revealing underlying data patterns, offering the potential for improved model performance. Therefore, we also hope to increase the divergence among those of boundary nodes. As shown in Figure 3, given a positive pair of the user u (like u_1) and the clicked item i (i_3), we first sort other neighbor item nodes i' (i_4, i_5) based on the distance metric between two items i and i'. Items with lower distances are considered to have higher correlation (similar nodes s_{im}) and nodes with higher distances are considered marginal nodes m_{ar}. The numbers of similar nodes and marginal nodes are set as K_1 and K_2. Then, the additional loss term is defined as follows:

$$
\mathcal{L}_{LEC} =
$$
$$
\sum_{(u,i) \in N^+} \sum_{s_{im} \in S(i)} \sum_{m_{ar} \in M(i)} -\left(\omega_{u,s_{im}} \log \sigma\left(e_u^\top e_{s_{im}}\right) - \omega_{u,m_{ar}} \log \sigma\left(e_u^\top e_{m_{ar}}\right)\right),
$$
$$
\omega_{u,s_{im}} = \frac{1}{\sqrt{d_u + 1}\sqrt{d_{s_{im}} + 1}}, \quad \omega_{u,m_{ar}} = \frac{1}{\sqrt{d_u + 1}\sqrt{d_{m_{ar}} + 1}},
$$
$$
\tag{11}
$$

where N^+ denotes the positive pairwise training dataset, $S(i)$ is the set of similar nodes about node i, $M(i)$ is the set of marginal nodes about node i, and $\omega_{u,s_{im}}, \omega_{u,m_{ar}}$ are the normalization coefficients. In particular, when sorting the items, we filter out the items far enough away from the node i whose distance is above a threshold θ to only preserve boundary nodes inspired by [Wang *et al.*, 2021c]. Here, we apply the co-occurrence similarity as the distance metric between two items, which is proportional to the number of users linked to both two items. Note that, the co-occurrence similarity is consistent with the idea of collaborative filtering [Wang *et al.*, 2020a; Chen *et al.*, 2021]. Hence the loss can be effectively adopted and added to common GCN-based collaborative filtering models. The overall algorithm process of our LEC loss can be found in Appendix.

With our proposed LEC, we finally derive the following training objective for GCN-based recommendation models:

$$
\mathcal{L} = \mathcal{L}_{CF} + \lambda \mathcal{L}_{LEC},
\tag{12}
$$

where $\mathcal{L}_{CF}$ represents the original loss function of the GCN-

based recommendation model like BPR loss in Equation 5, and hyper-parameter λ is to balance the LEC loss.

3.3 Comparison with Existing Methods

Recently, several works in the field of GCNs have been proposed to alleviate the over-smoothing problem. Here, we discuss the relations between our approach with them to provide a more comprehensive understanding.

Residual Connection. Motivated by the success of ResNet [He *et al.*, 2016], the residual connection has been the common sense to make GNNs go deeper and perform better, like LT-OCF [Choi *et al.*, 2021], LightGCN [He *et al.*, 2020], SGC [Wu *et al.*, 2021], DeepGCN [Li *et al.*, 2019], and GC-NII [Chen *et al.*, 2020c]. Taking GCNII as an example, it incorporates initial node embeddings into the output of each layer, which can be succinctly summarized as:

$$\mathbf{e}^{(k)} = \hat{\mathbf{e}}^{(k)} + \alpha(\mathbf{e}^{(0)} - \hat{\mathbf{e}}^{(k)}), \tag{13}$$

where $\hat{\mathbf{e}}^{(k)}$ and $\mathbf{e}^{(0)}$ are defined similarly to that in Equation 9. Intuitively, like our GMP approach, GCNII also perturbs the original node embedding $\hat{\mathbf{e}}^{(k)}$ with another vector $(\mathbf{e}^{(0)} - \hat{\mathbf{e}}^{(k)})$ of the direction moving away from over-smoothing point. The illustration of detailed embedding vector updating can be found at Appendix, which is considered to be a hidden reason that GCNII can also prevent embeddings from excessively converging to the over-smoothing point.

Normalization-based Methods. A proven way to reduce the over-smoothing effect is to normalize node embeddings to preserve the distinctiveness of embeddings during training. EGNN [Zhou *et al.*, 2021a] and Parinorm [Zhao and Akoglu, 2019] measure over-smoothing using Dirichlet energy and Total Pairwise Squared Distance(TPSD) respectively and optimize a GCN within a constrained range of Dirichlet energy and TPSD to preserve the distinctiveness of node embeddings individually. Additionally, Differentiable Group Normalization (DGN) [Zhou *et al.*, 2020] and NodeNorm [Zhou *et al.*, 2021b] use node-wise normalization. Although they maintain the embedding distances between nodes to avoid excessive similarity among global node embeddings in the optimization process, these methods of optimizing node embeddings lack any directionality and can even introduce noise that affects the performance of shallow GCN models. DGR maintains node embedding distinctiveness while providing a clear direction, which is to move away from over-smoothing point. Moreover, its application to shallow GCN models also leads to substantial improvements while normalization-based methods lead to a performance decrease.

Topology-based Methods. Some other approaches propose to modify the topological structure in each message passing layer to alleviate over-smoothing by randomly dropping edges or nodes in graphs, such as DropEdge [Rong *et al.*, 2019] and AdaEdge [Chen *et al.*, 2020a]. However, those approaches also damage the local topological structure around each node to some extent, which is critical in recommendations for modeling collaborative signals. In contrast, our approach not only penalizes the tendency of node embeddings approximating overly to be similar, but also maintains the topological structure and similarities among neighbor nodes.

Consequently, our method yields a more stable performance enhancement for recommendation. Another topology-based model IMP-GCN [Liu *et al.*, 2021] groups users with similar interests based on topological relationships and conducts separate message passing for each group to alleviate over-smoothing problem, which has the similar main idea with our proposed LEC, encouraging similar nodes to have comparable embeddings on the local graph.

3.4 Complexity Analysis

Actually, our approach is easy-to-use and model-free, which can be exploited into various GCN-based recommendation systems with a limited complexity cost. Assuming the original LightGCN backbone, the computational complexity can be represented as: $\mathcal{O}((N_u + N_i)\bar{d}TK)$, where T is the dimensionality of node embeddings, K is the number of graph convolutional layers, and $\bar{d}$ is the average number of neighboring nodes. After applying our DGR, the computational complexity becomes: $\mathcal{O}((N_u + N_i)(\bar{d} + 1)TK)$. It can be observed that DGR only brings a limited linear increase in computational complexity. As for the space complexity, DGR only introduces additional $(N_i + N_u)T$ parameters into the original GCN backbone. Furthermore, our experiments will demonstrate significant and robust improvements in DGR across various datasets and recommendation systems.

4 Experiments

To verify the effectiveness of the proposed DGR, we conduct extensive experiments based on five public benchmark datasets and report detailed analysis results in the section.

4.1 Experimental Setup

Datasets and Evaluation. Following many recent GCN-based recommendation systems [He *et al.*, 2020; Mao *et al.*, 2021; Yu *et al.*, 2023], five public benchmark datasets are used in our paper including Gowalla, Yelp2018, Movie-Lens1M, Netflix and Douban-Book. These datasets vary in domains, scale, and density. The statistics of the datasets can be found in Appendix. We partitioned each dataset into training, and testing sets using a 4:1 ratio. Recall@K and Normalized Discounted Cumulative Gain (NDCG)@K are chosen as the evaluation metrics that are popular in the evaluation of GCN-based recommendations.

Baseline Models. To demonstrate the generalizability, we incorporate our DGR into five widely-use GCN-based recommendation models including LightGCN [He *et al.*, 2020], SGL [Wu *et al.*, 2021], SimGCL [Yu and Qin, 2020], XSimGCL [Yu *et al.*, 2023] and MixGCF [Huang *et al.*, 2021]. In addition, we include two traditional recommendation models, BPRMF [Rendle *et al.*, 2012] and NGCF [Wang *et al.*, 2019b]. The latest state-of-the-art models, IMP-GCN and LayerGCN [Zhou *et al.*, 2023], are also included, which effectively mitigate the over-smoothing promblem in recommendation systems.

Experimental Setting. The experimental implementation details can be found in Appendix.

Datasets	Gowalla		MovieLens1M		Douban-Book		Yelp2018		Netflix	
	R@20	N@20	R@20	N@20	R@20	N@20	R@20	N@20	R@20	N@20
BPRMF	0.1393	0.1155	0.2486	0.2727	0.1263	0.1032	0.0496	0.0400	0.1034	0.0779
NGCF	0.1602	0.1311	0.2594	0.2676	0.1509	0.1259	0.0593	0.0492	0.1056	0.0814
IMP-GCN	0.1837	0.1510	0.2793	0.3154	0.1599	0.1349	0.0649	0.0529	0.1091	0.0795
LayerGCN	0.1825	0.1502	0.2798	0.3085	0.1619	0.1363	0.0636	0.0515	0.1088	0.0787
LightGCN	0.1832	0.1542	0.2690	0.3014	0.1497	0.1257	0.0650	0.0532	0.0868	0.0615
LightGCN+DGR	0.1880	0.1556	0.2749	0.3065	0.1573	0.1308	0.0669	0.0545	0.0909	0.0652
Improv.	2.61%	0.92%	2.19%	1.69%	5.08%	4.06%	2.92%	2.44%	4.72%	6.02%
SGL	0.1785	0.1503	0.2342	0.2194	0.1511	0.1362	0.0672	0.0553	0.1054	0.0803
SGL+DGR	0.1823	0.1532	0.2414	0.2271	0.1625	0.1453	0.0695	0.0571	0.1105	0.0826
Improv.	2.13%	1.93%	3.07%	3.51%	7.55%	6.68%	3.43%	3.26%	4.84%	2.86%
SimGCL	0.1821	0.1528	0.2560	0.2835	0.1773	0.1562	0.0717	0.0592	0.1213	0.0882
SimGCL+DGR	0.1860	0.1573	0.2752	0.3091	0.1811	0.1601	**0.0747**	**0.0616**	0.1271	0.0943
Improv.	2.14%	2.95%	7.50%	9.03%	2.14%	2.50%	4.18%	4.05%	4.78%	6.92%
XSimGCL	0.1681	0.1388	0.2498	0.2669	0.1756	0.1590	0.0684	0.0562	0.1319	0.1014
XSimGCL+DGR	0.1811	0.1533	0.2778	0.3102	0.1828	0.1633	0.0726	0.0599	**0.1365**	**0.1069**
Improv.	7.73%	10.49%	11.21%	16.22%	4.10%	2.70%	6.14%	6.58%	3.49%	5.42%
MixGCF	0.1852	0.1579	0.2735	0.3025	0.1771	0.1606	0.0697	0.0571	0.1198	0.0858
MixGCF+DGR	**0.1890**	**0.1604**	**0.2934**	**0.3306**	**0.1922**	**0.1730**	0.0719	0.0592	0.1228	0.0900
Improv.	2.05%	1.56%	7.23%	9.29%	8.53%	7.72%	3.16%	3.68%	2.50%	4.90%

Table 1: Performance comparison of baseline models before and after integrating DGR.

Model	LightGCN		SGL		SimGCL		XSimGCL		MixGCF	
	R@20	N@20	R@20	N@20	R@20	N@20	R@20	N@20	R@20	N@20
Baseline Model	0.1497	0.1257	0.1511	0.1362	0.1773	0.1562	0.1756	0.1590	0.1771	0.1606
Model+GMP	0.1533	0.1306	0.1575	0.1418	0.1786	0.1590	0.1777	0.1599	0.1807	0.1623
Improv.	+2.40%	+3.90%	+4.24%	4.11%	+0.73%	+1.79%	+1.20%	+0.57%	+2.03%	+1.06%
Model+LEC	0.1522	0.1279	0.1511	0.1363	0.1790	0.1575	0.1814	0.1631	0.1914	0.1716
Improv.	+1.67%	+1.75%	+0.00%	+0.07%	+0.96%	+0.83%	+3.30%	+2.58%	+8.07%	+6.85%
Model+DGR	0.1573	0.1308	0.1625	0.1453	0.1811	0.1601	0.1828	0.1633	0.1922	0.1730
Improv.	+5.08%	+4.06%	+7.54%	+6.68%	+2.14%	+2.50%	+4.10%	+2.70%	+8.53%	+7.72%

Table 2: Performance comparison of GMP and LEC components in DGR on Douban-Book.

4.2 Overall Performance

Table 1 summarizes the comparison of baseline models before and after integrating the proposed DGR with the percentage of relative improvement at each metric. The main observations are as follows:

(1) The baseline models incorporating DGR all exhibit performance improvements on all datasets, while also achieving state-of-the-art results. It demonstrates the effectiveness of the DGR, which unlocks the potential of personalized recommendation for GCN-based recommendation systems by mitigating the over-smoothing problem. In the vast majority of cases, the five popular models with DGR significantly outperform competitive traditional and recent state-of-the-art models. Baselines and Baselines with DGR in Table 1 apply the same layers of GCNs, which demonstrates that our proposed DGR can significantly improve shallow GCN-based recommendation models.

(2) Compared to all other models, DGR yields the substantial improvement for XSimGCL and SimGCL among all datasets. In particular, XSimGCL with DGR achieves a remarkable increase of 11.21% in Recall@20 and 16.22% in NDCG@20 on the MovieLens1M. We believe that XSimGCL and SimGCL employ the approach of adding noise to node embeddings instead of using complex data augmentation for graph self-supervised learning. This method may not effectively leverage topological information in the local graph, and introducing noise makes the model directionless. In contrast, DGR enables them to maintain distinctiveness among all user and item embeddings under the guidance of global topology while paying closer attention to the local topological information of local user-item interactions, leading to the enhanced performance.

(3) Comparing the performance on different datasets, DGR exhibits the substantial average improvement on MovieLens1M among all baseline models, which has the highest density, at 2.7%. The DGR can improve recommendation performance remarkably by slowing down its tendency towards over-smoothing. Meanwhile, DGR also has stable im-
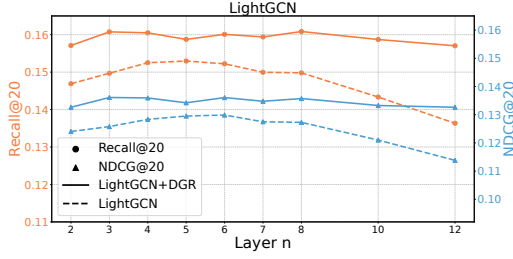
Figure 4: Performance of LightGCN with and without the DGR across a range of model depths. The orange line and blue line represent Recall@20 and NDCG@20, respectively. The solid line and dashed line represent models with and without DGR, respectively.

Model	LightGCN	SGL	SimGCL	XSimGCL	MixGCF
Baseline	2.68	2.61	2.65	2.32	2.38
Baseline+DGR	2.80	2.85	3.06	2.75	2.74

Table 3: **Row-diffs** of baseline models with and without DGR

Figure 5: Parameter Analysis of α based on LightGCN.

provements on sparse datasets such as Yelp2018 with 0.13% density. This is because DGR can effectively utilize the limited neighbors of each user (item) node to capture collaborative signals by learning on the local graph.

4.3 Further Analysis of DGR

In this section, we provide more experimental analysis of our proposed DGR for better understanding.

Ablation Study of DGR. We conduct the ablation experiments on Douban-Book by individually applying GMP, LEC, and DGR on different models. From Table 2, we can find that both GMP and LEC contribute to the improvements of model performance. The performance improvements provided by DGR is greater than the sum of the performance improvements from GMP and LEC, which implies that the synergistic combination of these two components is better suited for the desmoothing task in the context of recommendation. Additionally, we observe that GMP leads to a more remarkable performance improvement than LEC in LightGCN, SGL, and SimGCL. However, this trend is reversed in XSimGCL and MixGCF. We believe the primary reason is that the first three models are already capable of effectively capturing collaborative information during the aggregation of neighboring items(users), which makes global over-smoothing their performance bottleneck. Meanwhile, the latter two models are perceived as lacking the ability to capture collaborative signals from user-item interactions.

Case Study. The **Row-diff** is a metric to measure the degree of over-smoothing, proposed by [Zhao and Akoglu, 2019], where a lower row-diff score indicates a larger degree of over-smoothing. Its calculation formula is as follows:

$$\textbf{Row-diff}\left(\mathbf{E}^{(k)}\right) = \frac{1}{n^2} \sum_{i,j \in [n]} \left\| \mathbf{e}_i^{(k)} - \mathbf{e}_j^{(k)} \right\|_2, \quad (14)$$

where $\mathbf{E}^{(k)}$ is the node embedding matrix obtained after training k layers of GCN until convergence. In order to validate the ability of DGR to mitigate over-smoothing, we separately train baseline models and baseline models with DGR until convergence to obtain a converged embedding matrix. The row-diffs of node embeddings are computed respectively and shown in Table 3. It is evident that the application of DGR to the baseline models leads to a significant reduction in the degree of over-smoothing.
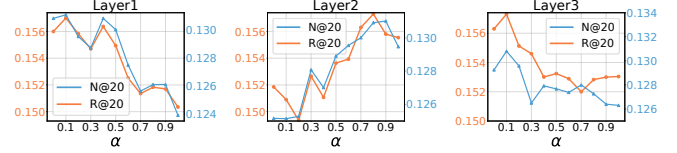
Impact of Model Depth. Although the DGR significantly enhances the shallow GCN-based recommendations as shown in Table 1, further exploration is needed to understand its desmoothing ability in deep GCN-based recommendations. We evaluate the five models with and without DGR across a range of model depths. We conduct these experiments on the Douban-Book dataset. The results of LightGCN are shown in Figure 4 while the results of other models can be found in Appendix. It is notable that models with DGR display a slower decline in performance as the number of layers increases, in contrast to those without DGR. This observation suggests that DGR possesses a robust desmoothing capability, which can effectively alleviate the detrimental effects of over-smoothing on recommendation performance in deep models.

Impact of hyper-parameters. As mentioned in Equation 9, the hyper-parameter α determines the extent to which all embeddings deviate from the over-smoothing point. It has a significant impact on the model performance. Here, We take LightGCN as the representation example to incorporate our DGR model with varying α values on Douban-Book dataset as shown in Figure 5, while the results of other models can be found in Appendix. The optimal value of α varies across different layers. Therefore, we explore the optimal α separately for each layer across a range of values, i.e., [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0].

5 Conclusion

In this work, we proposed a general and easy-to-use **D**esmoothing Framework for **G**CN-based **R**ecommendation Systems(**DGR**), to alleviate the over-smoothing problem in GCN-based recommendation systems, thereby unlocking the potential for personalized recommendation. We approached this problem from both global and local perspectives. Specifically, with the guidance of the global structure, we introduced **G**lobal Desmoothing **M**essage **P**assing(**GMP**), which penalizes the tendency of node embeddings approximating overly to be similar. In addition, we introduced **L**ocal Node **E**mbedding **C**orrection (**LEC**) for the readout embeddings to preserve the local collaborative relations between users and their neighboring items in the local graph. Extensive experiments based on five public datasets and five popular GCN-based recommendation models have demonstrated the effectiveness of our proposed DGR.

Acknowledgements

References

[Abbasnejad *et al.*, 2020] Ehsan Abbasnejad, Damien Teney, Amin Parvaneh, Javen Shi, and Anton van den Hengel. Counterfactual vision and language learning. In *Proc. of CVPR*, 2020.

[Cai *et al.*, 2023] Xuheng Cai, Chao Huang, Lianghao Xia, and Xubin Ren. Lightgcl: Simple yet effective graph contrastive learning for recommendation. *arXiv preprint arXiv:2302.08191*, 2023.

[Chen *et al.*, 2020a] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *Proc. of AAAI*, 2020.

[Chen *et al.*, 2020b] Lei Chen, Le Wu, Richang Hong, Kun Zhang, and Meng Wang. Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach. In *Proc. of AAAI*, 2020.

[Chen *et al.*, 2020c] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *Proc. of ICML*, 2020.

[Chen *et al.*, 2021] Ming Chen, Yunhao Li, and Xiuze Zhou. Conet: Co-occurrence neural networks for recommendation. *Future Generation Computer Systems*, 2021.

[Chen *et al.*, 2022a] Huiyuan Chen, Chin-Chia Michael Yeh, Fei Wang, and Hao Yang. Graph neural transport networks with non-local attentions for recommender systems. In *Proceedings of the ACM Web Conference 2022*, 2022.

[Chen *et al.*, 2022b] Liyi Chen, Zhi Li, Weidong He, Gong Cheng, Tong Xu, Nicholas Jing Yuan, and Enhong Chen. Entity summarization via exploiting description complementarity and salience. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[Chen *et al.*, 2022c] Liyi Chen, Zhi Li, Tong Xu, Han Wu, Zhefeng Wang, Nicholas Jing Yuan, and Enhong Chen. Multi-modal siamese network for entity alignment. In *Proc. of KDD*, 2022.

[Choi *et al.*, 2021] Jeongwhan Choi, Jinsung Jeon, and Noseong Park. Lt-ocf: learnable-time ode-based collaborative filtering. In *Proc. of CIKM*, 2021.

[Goyal *et al.*, 2019] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *Proc. of ICML*, 2019.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. of CVPR*, 2016.

[He *et al.*, 2020] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *Proc. of SIGIR*, 2020.

[He *et al.*, 2023] Li He, Xianzhi Wang, Dingxian Wang, Haoyuan Zou, Hongzhi Yin, and Guandong Xu. Simplifying graph-based collaborative filtering for recommendation. In *Proc. of WSDM*, 2023.

[Huang *et al.*, 2021] Tinglin Huang, Yuxiao Dong, Ming Ding, Zhen Yang, Wenzheng Feng, Xinyu Wang, and Jie Tang. Mixgcf: An improved training method for graph neural network-based recommender systems. In *Proc. of KDD*, 2021.

[Li *et al.*, 2019] Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In *Proc. of ICCV*, 2019.

[Li *et al.*, 2024] Hanzhe Li, Jingjing Gu, Xinjiang Lu, Dazhong Shen, Yuting Liu, YaNan Deng, Guoliang Shi, and Hui Xiong. Beyond relevance: Factor-level causal explanation for user travel decisions with counterfactual data augmentation. *ACM Transactions on Information Systems*, 2024.

[Lin *et al.*, 2022] Zihan Lin, Changxin Tian, Yupeng Hou, and Wayne Xin Zhao. Improving graph collaborative filtering with neighborhood-enriched contrastive learning. In *Proceedings of the ACM Web Conference 2022*, 2022.

[Liu *et al.*, 2021] Fan Liu, Zhiyong Cheng, Lei Zhu, Zan Gao, and Liqiang Nie. Interest-aware message-passing gcn for recommendation. In *Proc. of WWW*, 2021.

[Mao *et al.*, 2021] Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. Ultragcn: ultra simplification of graph convolutional networks for recommendation. In *Proc. of CIKM*, 2021.

[Peng *et al.*, 2022] Shaowen Peng, Kazunari Sugiyama, and Tsunenori Mine. Svd-gcn: A simplified graph convolution paradigm for recommendation. In *Proc. of CIKM*, 2022.

[Rendle *et al.*, 2012] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*, 2012.

[Rong *et al.*, 2019] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. *arXiv preprint arXiv:1907.10903*, 2019.

[Shuai *et al.*, 2022] Jie Shuai, Kun Zhang, Le Wu, Peijie Sun, Richang Hong, Meng Wang, and Yong Li. A review-aware graph contrastive learning framework for recommendation. In *Proc. of SIGIR*, 2022.

[Wang *et al.*, 2019a] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proc. of SIGIR*, 2019.

[Wang *et al.*, 2019b] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proc. of SIGIR*, 2019.

[Wang *et al.*, 2020a] Menghan Wang, Yujie Lin, Guli Lin, Keping Yang, and Xiao-ming Wu. M2grl: A multi-task multi-view graph representation learning framework for web-scale recommender systems. In *Proc. of KDD*, 2020.

[Wang *et al.*, 2020b] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. Disentangled graph collaborative filtering. In *Proc. of SIGIR*, 2020.

[Wang *et al.*, 2021a] Chao Wang, Hengshu Zhu, Qiming Hao, Keli Xiao, and Hui Xiong. Variable interval time sequence modeling for career trajectory prediction: Deep collaborative perspective. In *Proc. of WWW*, 2021.

[Wang *et al.*, 2021b] Chao Wang, Hengshu Zhu, Peng Wang, Chen Zhu, Xi Zhang, Enhong Chen, and Hui Xiong. Personalized and explainable employee training course recommendations: A bayesian variational approach. *ACM Transactions on Information Systems (TOIS)*, 2021.

[Wang *et al.*, 2021c] Zhenlei Wang, Jingsen Zhang, Hongteng Xu, Xu Chen, Yongfeng Zhang, Wayne Xin Zhao, and Ji-Rong Wen. Counterfactual data-augmented sequential recommendation. In *Proc. of SIGIR*, 2021.

[Wang *et al.*, 2023a] Chao Wang, Hengshu Zhu, Chen Zhu, Chuan Qin, Enhong Chen, and Hui Xiong. Setrank: A setwise bayesian approach for collaborative ranking in recommender system. *ACM Transactions on Information Systems*, 2023.

[Wang *et al.*, 2023b] Fei Wang, Zhi Zheng, Yongjun Zhang, Yansheng Li, Kun Yang, and Chenming Zhu. To see further: Knowledge graph-aware deep graph convolutional network for recommender systems. *Information Sciences*, 2023.

[Wang *et al.*, 2023c] Tianfu Wang, Li Shen, Qilin Fan, Tong Xu, Tongliang Liu, and Hui Xiong. Joint admission control and resource allocation of virtual network embedding via hierarchical deep reinforcement learning. *IEEE Transactions on Services Computing*, 2023.

[Wu *et al.*, 2021] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. Self-supervised graph learning for recommendation. In *Proc. of SIGIR*, 2021.

[Wu *et al.*, 2024] Wei Wu, Chao Wang, Dazhong Shen, Chuan Qin, Liyi Chen, and Hui Xiong. Afdgcf: Adaptive feature de-correlation graph collaborative filtering for recommendations. *arXiv preprint arXiv:2403.17416*, 2024.

[Xia *et al.*, 2022] Lianghao Xia, Chao Huang, Yong Xu, Jiashu Zhao, Dawei Yin, and Jimmy Huang. Hypergraph contrastive collaborative filtering. In *Proc. of SIGIR*, 2022.

[Xia *et al.*, 2023] Lianghao Xia, Chao Huang, Jiao Shi, and Yong Xu. Graph-less collaborative filtering. In *Proceedings of the ACM Web Conference 2023*, 2023.

[Yu and Qin, 2020] Wenhui Yu and Zheng Qin. Graph convolutional network for recommendation with low-pass collaborative filters. In *Proc. of ICML*, 2020.

[Yu *et al.*, 2023] Junliang Yu, Xin Xia, Tong Chen, Lizhen Cui, Nguyen Quoc Viet Hung, and Hongzhi Yin. Xsimgcl: Towards extremely simple graph contrastive learning for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[Zhao and Akoglu, 2019] Lingxiao Zhao and Leman Akoglu. Pairnorm: Tackling oversmoothing in gnns. *arXiv preprint arXiv:1909.12223*, 2019.

[Zhou *et al.*, 2020] Kaixiong Zhou, Xiao Huang, Yuening Li, Daochen Zha, Rui Chen, and Xia Hu. Towards deeper graph neural networks with differentiable group normalization. *Proc. of NeurIPS*, 2020.

[Zhou *et al.*, 2021a] Kaixiong Zhou, Xiao Huang, Daochen Zha, Rui Chen, Li Li, Soo-Hyun Choi, and Xia Hu. Dirichlet energy constrained learning for deep graph neural networks. *Proc. of NeurIPS*, 2021.

[Zhou *et al.*, 2021b] Kuangqi Zhou, Yanfei Dong, Kaixin Wang, Wee Sun Lee, Bryan Hooi, Huan Xu, and Jiashi Feng. Understanding and resolving performance degradation in deep graph convolutional networks. In *Proc. of CIKM*, 2021.

[Zhou *et al.*, 2023] Xin Zhou, Donghui Lin, Yong Liu, and Chunyan Miao. Layer-refined graph convolutional networks for recommendation. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, 2023.