PolyCF: Towards Optimal Spectral Graph Filters for Collaborative Filtering

YIFANG QIN, School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU-Anker LLM Lab, Peking University, Beijing, China
WEI JU, College of Computer Science, Sichuan University, Chengdu, China
YIYANG GU, School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU-Anker LLM Lab, Peking University, Beijing, China
ZIYUE QIAO, School of Computing and Information Technology, Great Bay University, Dongguan, China
ZHIPING XIAO, Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, Washington, USA
MING ZHANG, School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU-Anker LLM Lab, Peking University, Beijing, China

Collaborative Filtering (CF) is a pivotal research area in recommender systems that capitalizes on collaborative similarities between users and items to provide personalized recommendations. With the remarkable achievements of node embedding-based Graph Neural Networks (GNNs), we explore the upper bounds of expressiveness inherent to embedding-based methodologies and tackle the challenges by reframing the CF task as a graph-signal processing problem. To this end, we propose PolyCF, a flexible graph signal filter that leverages polynomial graph filters to process interaction signals. PolyCF exhibits the capability to capture spectral features across multiple eigenspaces through a series of Generalized Gram filters and is able to approximate the optimal polynomial response function for recovering missing interactions. A graph optimization objective and a pairwise ranking objective are jointly used to optimize the parameters of the convolution kernel. Experiments on three widely adopted datasets demonstrate the superiority of PolyCF over the state-of-the-art CF methods.

CCS Concepts: • **Information systems** → **Recommender systems**;

Additional Key Words and Phrases: Collaborative Filtering, Graph Signal Processing

This article is partially supported by the National Key Research and Development Program of China under Grant No. 2023YFC3341203, the National Natural Science Foundation of China under Grant Nos. 62306014 and 62276002, the Sichuan Science and Technology Program under Grant No. 2025ZNSFSC1506, and the Sichuan University Interdisciplinary Innovation Fund.

Authors' Contact Information: Yifang Qin, School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU-Anker LLM Lab, Peking University, Beijing, China; e-mail: qinyifang@pku.edu.cn; Wei Ju (corresponding author), College of Computer Science, Sichuan University, Chengdu, China; e-mail: juwei@scu.edu.cn; Yiyang Gu, School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU-Anker LLM Lab, Peking University, Beijing, China; e-mail: yiyanggu@pku.edu.cn; Ziyue Qiao, School of Computing and Information Technology, Great Bay University, Dongguan, China; e-mail: ziyuejoe@gmail.com; Zhiping Xiao (corresponding author), Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, Washington, USA; e-mail: patxiao@uw.edu; Ming Zhang (corresponding author), School of Computer Science, State Key Laboratory for Multimedia Information Processing, PKU-Anker LLM Lab, Peking University, Beijing, China; e-mail: mzhang_cs@pku.edu.cn.

ACM 1558-2868/2025/6-ART94
https://doi.org/10.1145/3728464

ACM Reference format:
Yifang Qin, Wei Ju, Yiyang Gu, Ziyue Qiao, Zhiping Xiao, and Ming Zhang. 2025. PolyCF: Towards Optimal Spectral Graph Filters for Collaborative Filtering. *ACM Trans. Inf. Syst.* 43, 4, Article 94 (June 2025), 28 pages. https://doi.org/10.1145/3728464

1 Introduction

Collaborative Filtering (CF) stands out as one of the most popular research topics of recommender systems, offering an effective solution to the issue of information-overload in a wide range of web applications [8, 26, 53]. The primary goal of CF tasks is to recommend the most suitable items for each user to interact with, based on their historical interactions. Early CF methods have traditionally framed the CF task as a matrix factorization problem [19, 33, 41]. Subsequent research has emphasized that the crux of solving CF problems lies in modeling the collaborative affinity between users and items according to their interaction history [11, 28].

With the rapid advancement of **Graph Neural Networks (GNNs)** and their notable success in graph representation and a series of practical downstream tasks [9, 22, 29, 32, 69], an increasing number of research endeavors have shifted their focus toward applying graph-based methods to CF. Early works such as NGCF [54] and LightCGN [26] propose to integrate **Graph Convolution Networks (GCNs)** into interaction graphs to capture high-order connectivity between users and items. These graph-based models map the nodes into embedding vectors and fully exploit neighborhood structures through a message-passing paradigm. Inspired by these efforts, later graph-based models have further enhanced the model performance and expressiveness of graph-embedding-based CF methods.

For instance, DGCF [55] introduces disentangled graph convolution to capture the rich semantics of interactions. UltraGCN [40] extends the propagation function of LightGCN to an infinite number of layers while simplifying the computation process. JGCF [21] focuses on applying trainable linear polynomial graph convolution to capture spectral features. Other works concentrate on enhancing the learning of more representative node embeddings through contrastive-learning methods. SGL [61] proposes to learn informative node representations through contrastive learning between original and augmented graphs. LightGCL [3] further enhances the contrastive view with the **Singular Value Decomposition (SVD)** of the interaction graphs, offering a spectral perspective.

Despite the prosperity of node embedding-based graph CF models, there are increasing research works revealing the limitations of such methods and using alternative approaches to achieve state-of-the-art performance. EASE^{R} [51] discusses the closed-form solution of the training objective for a linear encoder and achieves unexpectedly promising results with a shallow one-layer filter. Later works focus on obtaining powerful graph filters and tackling CF problems using **Graph Signal Processing (GSP)**. GF-CF [49] and PGSP [36] explore the effectiveness of graph low-pass filters as a universal solution for various CF tasks. GS-IMC [4] introduces a class of regularization functions to address CF as an inductive one-bit matrix completion task. However, it is essential to note that existing GSP methods heavily rely on manually crafted graph filters, which are usually derived from empirical observations made with previous CF models that may inevitably introduce inductive biases into the model. Furthermore, the theoretical analysis of these utilized filters is currently limited to symmetrical-normalized Laplacian matrices, which consequently restricts the potential of the adopted graph filters.

To address the aforementioned challenges of graph-based CF models, our work starts with a comprehensive analysis of the upper bounds of expressiveness achievable by node embedding-based graph models. From this analysis, we uncover the fundamental connection between traditional

embedding-based method and low-rank matrix factorization. Consequently, we demonstrate that the performance of graph embedding models is inherently constrained by their limited embedding dimensions, thereby highlighting the potential advantages of graph filter-based methods over graph embeddings. Subsequently, we introduce the rationale behind a novel graph filter–based approach, denoted as PolyCF, which goes beyond limitations imposed by the embedding size of nodes. We equip PolyCF with a novel generalized normalization of the Gram matrix as the filter backbone, enabling it to capture spectral characteristics originating from diverse eigenspace structures. A learnable polynomial convolution kernel empowers the generalized Gram kernel with the capability of approximating the optimal filter function tailored to the specific scenario. A graph optimization target function and a pairwise training objective are jointly employed to optimize the model parameters. Comparative experiments against a variety of state-of-the-art CF methods conclusively demonstrate the superiority of the proposed PolyCF. In summary, the principal contributions of this work can be summarized and listed as follows:

- —We conduct a comprehensive analysis of the expressiveness potential of existing node embedding-based models and introduce PolyCF, a state-of-the-art graph filter-based method capable of approximating the optimal response function for CF problems.
- —We propose the generalized normalization of Gram matrices, empowering the model to capture various spectral characteristics using a set of generalized Gram filters.
- —Extensive experiments on three real-world recommendation datasets have validated the effectiveness of PolyCF. Further studies and analysis have revealed the functionality of PolyCF and the generalization capability of the model parameters.

2 Related Works

2.1 CF

CF tasks aim to recommend a personalized set of items to users based on similarities derived from their past interaction history [7, 18, 57, 70]. Traditional CF methods typically frame CF as a matrix factorization problem [33, 41]. In contrast, more recent ranking-based approaches have concentrated on minimizing pairwise ranking loss [39, 48, 52]. Another widely explored class of methods focuses on modeling the transition relationships between items [1, 20, 27] with random processes such as Markov Chain.

The growing interest in GNNs [30, 31, 38] and the application of graph-based methodologies [15, 66] has led recent research in recommender systems to widely adopt graph-based models, such as sequence recommendation [44, 46, 62], social networks [12, 13], and location-based recommendation [45, 58]. The message-passing structure of GNNs empowers them to capture the topological structures and exploit implicit similarities present in multi-hop neighborhoods of nodes [65, 67]. Among the notable contributions in this area, NGCF [54] and LightGCN [26] stand as pioneering work that introduces graph convolution into CF problems. Later works like UltraGCN [40] and CAGCN [59] seek refinement of the message function of GCNs. There are also researches like CGI [60], GraphDA [14], RocSE [67], and ClusterGCN [35] that improve recommendation performance with graph structure learning techniques. Since the progress in graph self-supervised learning, works like XSimGCL [68], CGCL [25], and MPT [23] have been proposed to integrate unsupervised learning objectives into a variety of recommendation tasks.

2.2 GSP

In addition to the traditional message-passing scheme employed by GNNs, there has been a significant focus on GSP methods, which examine the filtering properties of graph operators from a spectral perspective [10, 37, 47]. In a typical GSP model, input node features are treated as

graph signals, and filters derived from the graph topology are applied to process these signals. Inspired by the pioneering work in GCNs [9], cutting-edge research in the field of GNNs has unveiled the potential of high-order polynomial graph filters for expressing arbitrary smooth filter functions [24, 56].

Inspired by the success of GSP, there have been attempts to integrate the capabilities of graph filters into CF models. Graph filter-based models like EASE^{R} [51], GF-CF [49], PGSP [36], and SGFCF [43] directly process interaction signals from the rows of the interaction matrix and achieve promising results in various CF tasks. FIRE [63] proposes to incorporate graph filters into incremental recommendations. However, it's important to note that existing GSP methods still heavily rely on manually crafted filter structures, which do not fully harness the potential of graph filters.

3 Preliminary

In this section, we will provide a concise overview of graph-based CF and introduce the notations used in GSP for clarity in the following sections.

3.1 Graph-Based CF

A typical CF task is conceptualized as a matrix completion problem that requires recovering a matrix with missing values. Specifically, the observed binary interaction matrix $R \in \{0,1\}^{m\times n}$ is derived from an interaction system involving m users and n items, where $R_{i,j} = 1$ signifies the observation of an interaction between user i and item j. The objective of the CF task is to acquire the reconstructed matrix $R^* \in 0, 1^{m\times n}$ for augmenting R with additional interactions that match similar users and items according to the collaborative affinity obtained from original matrix R.

For graph-based methods, the CF task is further expanded to a link prediction task. Specifically, the corresponding user–item bipartite graph is formulated with the adjacency matrix $A = \begin{bmatrix} 0 & R \\ R^T & 0 \end{bmatrix}$. The majority of graph-based methods adapt normalized graph convolution [32] to propagate messages on the bipartite graph. Specifically, the normalized interaction matrix is calculated with a row- and column-wise normalization of R:

$$\tilde{R} = D_U^{-\frac{1}{2}} R D_I^{-\frac{1}{2}}, \tag{1}$$

to balance the edge weights according to node degree. Similarly, the normalized adjacency matrix $\tilde{A} = \begin{bmatrix} 0 & \tilde{R} \\ \tilde{R}^T & 0 \end{bmatrix}$. In practice, another commonly used graph is the item's Gram graph, represented by its adjacency matrix denoted as $\tilde{G}_I = \tilde{R}^T\tilde{R}$, which captures the collaborative relationships among items. Similarly, the user's Gram matrix is formulated as $\tilde{G}_U = \tilde{R}\tilde{R}^T$.

3.2 GSP

Given a weighted graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ represents the node set and $\mathcal{E}$ represents the edge set, the Laplacian matrix L of $\mathcal{G}$ is defined with $L = D - A$, where $D = diag\{d_1, ..., d_n\}$ is diagonal degree matrix. Specifically, the graph signals is presented as $x \in \mathbb{R}^{|\mathcal{V}|}$, with each component x_i representing the signal response at node i. Graph filters are defined as mappings on graph signals and depend on the topological structure of $\mathcal{G}$.

To characterize the smoothness of graph signals, we formulate the graph derivative of a given signal x using the graph quadratic form [50], formulated as follows:

$$S_{\mathcal{G}}(x) = x^T L x. \tag{2}$$

The smoothness of graph signals on a given graph structure reflects the overall differences between adjacent nodes.

As the Laplacian matrix L is positive semi-definite, it can be further factorized through its eigendecomposition: $L = U^T \Lambda U$, where $\Lambda = diag\{\lambda_1, ..., \lambda_{|\mathcal{V}|}\}$ is a diagonal matrix that is composed of eigenvalues of L, $U = [u_1, ..., u_{|\mathcal{V}|}]$ represents a set of corresponding normalized orthogonal eigenvectors. Graph filters are typically constructed based on the Graph Fourier Transform $\hat{x} = Ux$, which maps the graph signal x into the graph eigenspace:

$$f_{\mathcal{G}}(x) = U^T diag\{f(\lambda_1), ..., f(\lambda_n)\}Ux. \tag{3}$$

With the development of spectral graph theory, recent research has increasingly focused on leveraging graph filter methods to construct CF models. While many of these methods have achieved promising performance, there is currently no approach grounded in a theoretical analysis of the expressiveness of graph filters compared to graph embedding methods. This gap limits the development of more flexible and expressive filter-based models.

4 Methodology

In this section, we will start with conducting a brief review of the current node embedding-based graph methodologies and discuss their capacity limits. After that, we will introduce the proposed approach, namely *PolyCF*. As illustrated in Figure 2, PolyCF employs a multi-channel graph convolution operator, that comprises a series of generalized Gram convolution kernels. Finally, we will elucidate the optimization methodology employed to acquire the optimal polynomial approximation graph filters for CF.

4.1 From Embedding to Filter-Based Models

Numerous graph-based CF methods have emerged, employing node embeddings to represent users and items, and subsequently, to compute recommendation scores. Nevertheless, the full range of expressive capabilities and constraints associated with these node embedding-based methods remains to be explored. As an illustration, we consider the polynomial graph convolution model and dive into its capacity of recovering missing interactions. An illustrative comparison between two types of methods will be discussed and is presented in Figure 1.

To formulate a standard embedding-based method, like LightGCN [26] and JGCF [21], it is parameterized by an embedding matrix $E = \left[E_U, E_I\right] \in \mathbb{R}^{(m+n)\times d}$, where E_U, E_I represent node embeddings for users and items, respectively. The propagation process can then be expressed as a summation of layer outputs that follows:

$$P(\tilde{A})E = \sum_{k=0}^{K} \alpha_k \tilde{A}^k E, \tag{4}$$

where α_ks denotes the coefficients of the polynomial graph filter. More specifically, the reconstructed interaction matrix is derived from the inner-product similarity between the propagated embeddings of users and items:

$$R^* = [P(\tilde{A})E]_{:m} \cdot [P(\tilde{A})E]^T_{m+1:}. \tag{5}$$

To explore the expressiveness of the resulting reconstructed matrix in Equation (5), we conduct a more in-depth analysis of the node embeddings obtained through the widely applied graph convolution layers. Specifically, for arbitrary polynomial graph convolutional models without layer activation functions, we derive the following theorem to reveal the limitations of such node embedding-based methods.

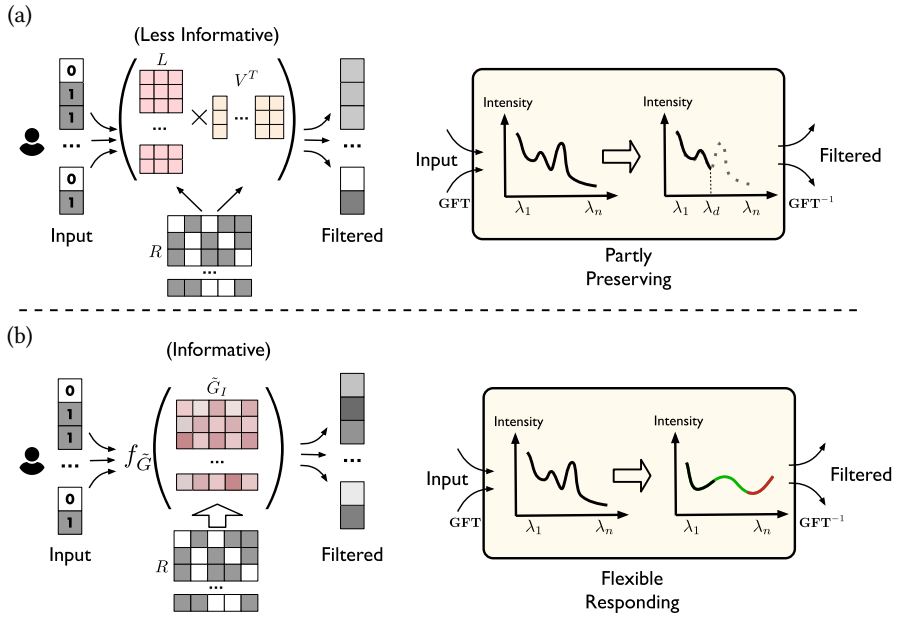


Fig. 1. Comparison between node embedding-based method (a) and graph filter-based methods (b). The filtering process of both approaches is illustrated on the left, while the corresponding spectral responding behaviors are presented on the right. The embedding-based methods lead to spectral information loss due to its nature of low-rank factorization, while the filter-based methods offer greater flexibility, enabling diverse responses such as band-pass smoothing (represented by the green curve) and high-pass enhancement (represented by the red curve).

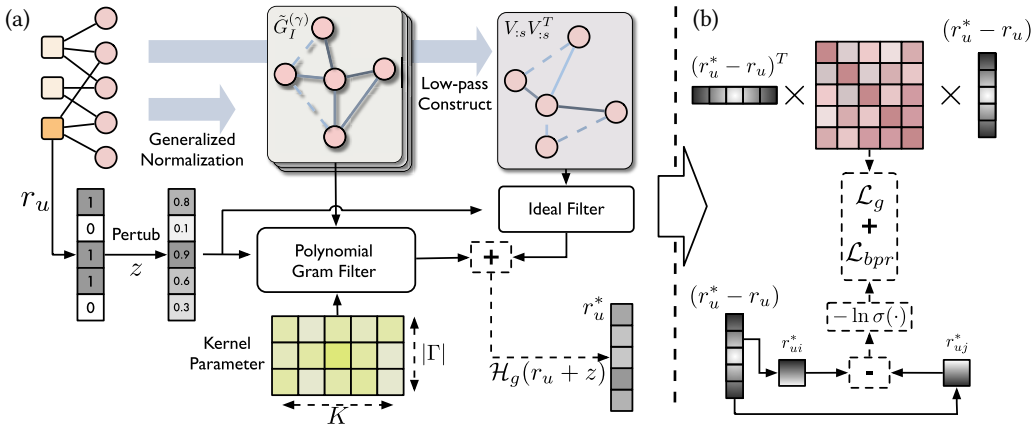


Fig. 2. A general illustration of the training framework of PolyCF.

Theorem 4.1. *For any polynomial graph filter $P(\tilde{A})$, there exists $L, V \in \mathbb{R}^{n \times d}$ that satisfy the relationship:*

$$R^* = \tilde{R} \cdot LV^T, \tag{6}$$

where both L and V are linear combinations of E_I and $\tilde{R}E_U$. The computation of these two matrices depends solely on the polynomial coefficients α_k and the item Gram matrix $\tilde{G}_I$.

Proof. Notice that the adjacency $\tilde{A}$ is a block matrix and its kth power could be rewritten as:

$$\tilde{A}^{2n+1}E = \begin{bmatrix} \tilde{G}_U & \\ & \tilde{G}_I \end{bmatrix}^n \begin{bmatrix} & \tilde{R} \\ \tilde{R}^T & \end{bmatrix} \begin{bmatrix} E_U \\ E_I \end{bmatrix} = \begin{bmatrix} \tilde{G}_U^n \tilde{R} E_I \\ \tilde{G}_I^n \tilde{R}^T E_U \end{bmatrix} \tag{7}$$

$$\tilde{A}^{2n}E = \begin{bmatrix} \tilde{G}_U & \\ & \tilde{G}_I \end{bmatrix}^n \begin{bmatrix} E_U \\ E_I \end{bmatrix} = \begin{bmatrix} \tilde{G}_U^n E_U \\ \tilde{G}_I^n E_I \end{bmatrix}, \tag{8}$$

where $\tilde{G}_U, \tilde{G}_I$ represents Gram matrix of users and items. The reconstructed matrix in Equation (5) can thus be formulated with:

$$\begin{aligned} R^* &= [(P_{2k}(\tilde{A}) + P_{2k+1}(\tilde{A}))E]_{:m} \cdot [(P_{2k}(\tilde{A}) + P_{2k+1}(\tilde{A}))E]^T_{m+1:} \\ &= [P_{2k}(\tilde{G}_U)E_U + P_{2k+1}(\tilde{G}_U)\tilde{R}E_I] \cdot [P_{2k}(\tilde{G}_I)E_I + P_{2k+1}(\tilde{G}_I)\tilde{R}^T E_U]^T \end{aligned} \tag{9}$$

Notice that the polynomial of user Gram matrix $\tilde{G}_U$s can be transferred into polynomial of $\tilde{G}_I$s via:

$$\tilde{G}_U^n = \tilde{R}\tilde{G}_I^{n-1}\tilde{R}^T, \ \forall n > 0. \tag{10}$$

The reconstructed R^* can be further formulated as:

$$\begin{aligned} R^* &= [\tilde{R}P_{2k}^{(-1)}(\tilde{G}_I)\tilde{R}^T E_U + \tilde{R}P_{2k+1}(\tilde{G}_I)E_I] \cdot [P_{2k}(\tilde{G}_I)E_I + P_{2k+1}(\tilde{G}_I)\tilde{R}^T E_U]^T \\ &= \tilde{R}[\sum_{k=1}^{\lfloor \frac{K}{2}-1 \rfloor} \tilde{G}_I^{2k-1}(\alpha_{2k+1}E_I + \alpha_{2k}\tilde{R}^T E_U)] \cdot [\sum_{k=0}^{\lfloor \frac{K}{2} \rfloor} \tilde{G}_I^{2k}(\alpha_{2k}E_I + \alpha_{2k+1}\tilde{R}^T E_U)]^T \\ &= \tilde{R} \cdot LV^T, \end{aligned} \tag{11}$$

where the linear combinations L and V are obtained as:

$$L = \sum_{k=1}^{\lfloor \frac{K}{2}-1 \rfloor} \tilde{G}_I^{2k-1}(\alpha_{2k+1}E_I + \alpha_{2k}\tilde{R}^T E_U), \ V = \sum_{k=0}^{\lfloor \frac{K}{2} \rfloor} \tilde{G}_I^{2k}(\alpha_{2k}E_I + \alpha_{2k+1}\tilde{R}^T E_U). \tag{12}$$

Consequently, we draw the following crucial corollary from Theorem 4.1:

Corollary 4.2. *For arbitrary embedding-based model (e.g,. LightGCN) that propagates node embeddings with polynomial graph convolution, there exists a corresponding matrix R' s.t. the d-order low-rank factorization of R' is equivalent to the model's learned embeddings. Here R' is also a polynomial transformation of R that is determined by the propagation scheme.*

Theorem 4.1 and Corollary 4.2 reveal the inherent relationship between typical embedding-based and low-rank factorization methods. Specifically, for a degenerated case $P(\tilde{A}) = I$, the learned embedding is equivalent to a classical MF model [5, 33]. Similarly, for an n-layer LightGCN [26] with the propagation of $P(\tilde{A}) = \frac{1}{n}\sum_{l=0}^{n}\tilde{A}^l$, we can also express it with a low-rank decomposition that based on the polynomial combination of $\tilde{G}_I$ and $E_I + \tilde{R}E_U$:

$$R^*_{\text{LightGCN}} = \frac{1}{n^2}\tilde{R}[\sum_{k=1}^{\lfloor \frac{K}{2}-1 \rfloor} \tilde{G}_I^{2k-1}(E_I + \tilde{R}E_U)] \cdot [\sum_{k=1}^{\lfloor \frac{K}{2} \rfloor} \tilde{G}_I^{2k}(E_I + \tilde{R}E_U)]^T \tag{13}$$

From these observations, we can draw several key insights regarding node embedding-based methods:

— *Expressiveness Limitations of Node Embeddings*: The expressiveness of node embedding-based graph methods is inherently constrained by the embedding size d. This limitation stems from the fact that the rank of the reconstructed matrix R^* is bounded by $\min\{rank(U), rank(V)\} \leq d$. As a result, their ability to recover missing interactions is fundamentally restricted by the predefined embedding size d. As illustrated in Figure 1(a), this constraint implies the model could only preserve the top-d eigenvalue information from the interaction matrix at most, while the high-frequency components are neglected and cannot be fully utilized.
— The normalized item Gram matrix $\tilde{G}_I$ is the key to overcome the limitations due to the node embedding size. Specifically, the minimum rank of $\tilde{G}_I$ can be bounded by Sylvester inequality:

$$rank(\tilde{G}_I) \geq 2rank(\tilde{R}) - m \approx m. \tag{14}$$

where m represents the number of users. In practice, m is significantly larger than feasible embedding size d. As illustrated in Figure 1(a), transitioning from a factorization-based formulation to filtering transformations based on $\tilde{G}_I$ can substantially enhance the retained eigenspace features, therefore incorporating additional information.

Motivated by the insights gleaned from the aforementioned observations, we introduce the proposed graph filter-based PolyCF, which doesn't require node embeddings and is built upon stacking layers of the generalized normalization of Gram convolution kernel.

4.2 Generalized Gram Convolution

Recall that the normalized item Gram matrix is derived from the product of the normalized interactions, given by $\tilde{G}_I = \tilde{R}^T\tilde{R} \in \mathbb{R}^{n\times n}$. This particular form of the Gram matrix has been widely adopted to design graph filters in prior studies [36, 49, 51] for capturing item co-occurrence patterns. Despite its proven effectiveness in practical applications, its characteristics as a graph filter have remained unexplored. Therefore, we undertake a comprehensive analysis from a graph spectral perspective to develop a more potent polynomial Gram filter for our proposed PolyCF.

4.2.1 Generalized Gram Filter. In previous works, the Laplacian matrices are often symmetrically normalized, which constrained the expressiveness of the associated graph filter. Studies on graph convolution [17] have indicated that employing random-walk normalized matrices, akin to PageRank [2] could yield more personalized propagation outcomes. We extend the idea to encompass more flexible situations and propose the notion of *Generalized Normalization*. For any given adjacency matrix A, the generalized normalization of its Laplacian matrix is defined as follows:

$$\tilde{L}^{(\gamma)} = D^{-\gamma}(D - A)D^{\gamma-1} = I - D^{-\gamma}AD^{\gamma-1}. \tag{15}$$

When $\gamma = \frac{1}{2}$, this generalized normalization is equivalent to symmetrical normalization, and when $\gamma = 1$ it becomes random-walk normalization. Similarly, the generalized normalization of gram matrix G_I is formulated as:

$$\tilde{G}_I^{(\gamma)} = \tilde{R}^{(-\gamma)T}\tilde{R}^{(-\gamma)} = D_I^{-\gamma}R^T D_U^{-1}RD_I^{\gamma-1}. \tag{16}$$

By introducing the concept of generalized normalization for G_I, we can leverage its adaptable spectral structure.

Theorem 4.3. *For any $\gamma \in [0, 1]$, the Laplacian of $\tilde{G}_I^{(\gamma)}$ shares the same set of eigenvalues that satisfies the following inequality relationship:*

$$0 \leq \lambda_1 \leq \ldots \leq \lambda_n \leq 1. \tag{17}$$

Moreover, for any $\mu_i^{(\gamma_1)}$ and $\mu_i^{(\gamma_2)}$ are arbitrary eigenvectors associated with eigen value λ_i for $\tilde{G}_I^{(\gamma_1)}$ and $\tilde{G}_I^{(\gamma_2)}$ respectively, the relationship holds:

$$\mu_i^{(\gamma_1)} = D_I^{\gamma_1-\gamma_2}\mu_i^{(\gamma_2)} \tag{18}$$

Proof. First, we start with calculating the eigenvalues of the interaction adjacency when $\gamma = \frac{1}{2}$:

$$\tilde{A}^{(\frac{1}{2})} = \begin{bmatrix} D_U^{-\frac{1}{2}} & \\ & D_I^{-\frac{1}{2}} \end{bmatrix} \begin{bmatrix} & R \\ R^T & \end{bmatrix} \begin{bmatrix} D_U^{-\frac{1}{2}} & \\ & D_I^{-\frac{1}{2}} \end{bmatrix} \tag{19}$$

In this special case, it has been proven by previous research [32] that the eigenvalues of $\tilde{A}^{(\frac{1}{2})}$ are within $[-1, 1]$. Notice that the $\tilde{A}^2 = \begin{bmatrix} \tilde{G}_U^{(\frac{1}{2})} & \\ & \tilde{G}_I^{(\frac{1}{2})} \end{bmatrix}$ is a diagonal matrix, therefore the eigenvalues of $\tilde{G}_I^{(\frac{1}{2})}$ are within $[0, 1]$. For cases where $\gamma_1 \neq \frac{1}{2}$, given any $\lambda_i, \mu_i^{(\frac{1}{2})}$ are pair of eigenvalue and vector, we have:

$$\begin{aligned} \tilde{G}_I^{(\gamma_1)}(D_I^{\frac{1}{2}-\gamma_1}\mu_i^{(\frac{1}{2})}) &= D_I^{\frac{1}{2}-\gamma_1}\tilde{G}_I^{(\frac{1}{2})}D_I^{\gamma_1-\frac{1}{2}}(D_I^{\frac{1}{2}-\gamma_1}\mu_i^{(\frac{1}{2})}) \\ &= D_I^{\frac{1}{2}-\gamma_1}\tilde{G}_I^{\frac{1}{2}}\mu_i^{(\frac{1}{2})} \\ &= \lambda_i D_I^{\frac{1}{2}-\gamma_1}\mu_i^{(\frac{1}{2})}. \end{aligned} \tag{20}$$

The equation indicates that $\tilde{G}_I^{(\gamma_1)}$ shares the same eigenvalues with $\tilde{G}_I^{(\frac{1}{2})}$ and the eigenvectors can be transferred with:

$$\mu_i^{(\gamma_1)} = D_I^{\frac{1}{2}-\gamma}\mu_i^{(\frac{1}{2})}. \tag{21}$$

Similarly, we can conclude that $\tilde{G}_I^{(\gamma_2)}$ also shares the eigenvalues and the corresponding eigenvectors can be transformed with:

$$\mu_i^{(\gamma_1)} = D_I^{\frac{1}{2}-\gamma}\mu_i^{(\frac{1}{2})}. \tag{22}$$

By combining Equations (21) and (22), we can conclude that the eigenvectors of two generalized normalizations of G_I on the same eigenvalue can be transformed to each other with:

$$\mu_i^{(\gamma_1)} = D_I^{\gamma_1-\gamma_2}\mu_i^{(\gamma_2)}. \tag{23}$$

□

Theorem 4.3 implies that Gram matrices normalized with different values of γ share similar spectral spaces, while still maintaining their distinct eigenspaces. Compared to the regular symmetrized normalizations, the generalized form offers greater flexibility via different choices of normalization factors γ, thereby framing distinct spectral response functions.

4.2.2 Polynomial Graph Convolution. Having obtained the generalized Gram filters $\tilde{G}_I^{(\beta)}$, our next step is to devise a polynomial spectral filter based on them. To formulate, consider an arbitrary normalized Laplacian matrix $\tilde{L}$ with its eigendecomposition $\tilde{L} = U^T\Lambda U$. Our objective is to identify the possible form that expresses potential optimal linear filters that depict the transition between observed interactions and reconstructed interactions. More specifically, for a user with its interaction history $r_u \in \{0, 1\}^n$, the Gram filter is applied as:

$$r_u^* = h(\tilde{L})r_u = U^T h(\Lambda)Ur_u = U^T diag\{h(\lambda_1), ..., h(\lambda_n)\}Ur_u, \tag{24}$$

where the $h(\lambda) : \mathbb{R} \rightarrow \mathbb{R}$ corresponds to the response function which determines the behavior of the graph filter. Specifically, the graph filter $h(\tilde{L})$ operates the interaction vectors by decomposing the input signals into the eigenspace, and then recombining the spectral components based on $h(\lambda)$. Drawing inspiration from previous research on polynomial graph convolution [9, 24, 56], we express $h(\lambda)$ as a linear combination of a set of orthogonal polynomial basis functions:

$$h(\lambda) = \sum_{k=0}^{K} \theta_k \cdot P_k(1-\lambda), \tag{25}$$

where θ_k represents the linear coefficient and $P_k(\lambda)$ denotes the kth basis function, which incorporates a k-order polynomial of λ. Equation (25) defines a K-order filter function that can be applied to the spectral domain of the Gram item graph using $h(\tilde{G})$. There exist various options for selecting the polynomial basis, such as the Chebyshev basis and the Bernstein basis. Several of the orthogonal bases have proven to be expressive when fitting to specific response functions, such as linear low-pass filters [9] or more general bandpass filters [24, 56].

The derived generalized normalization of Gram matrix in Equation (16) enables the model to conduct graph filtering on an input signal using filters possessing distinct eigenspace structures. This capability empowers the resulting polynomial filter to leverage the rich spectral features of graph signals. In summary, the proposed generalized Gram convolution kernel is defined as:

$$\mathcal{H}_P(G_I) = \frac{1}{|\Gamma|} \sum_{\gamma \in \Gamma} \sum_{k=0}^{K} \theta_k^{(\gamma)} P_k(\tilde{G}_I^{(\gamma)}), \tag{26}$$

where $\theta_k^{(\gamma)}$ is the parameter of the generalized convolution kernel, which can be optimized using gradient-based methods. Γ represents the set of used generalized normalization orders. While Theorem 4.3 indicates that filters defined over a wide range of γ can capture a diverse set of spectral components, we extend this advantage by incorporating the polynomial filter basis. By adaptively aggregating the filtering outcomes from diversified eigenspaces, the generalized Gram convolution is capable of learning more comprehensive filter functions to recover missing interactions.

4.2.3 Low-Pass Enhancement. Despite the generalized Gram convolution kernel presented in Equation (26) offers a polynomial approximation for the desired graph filter, the optimal filter describing the transformation between R and R^* may not exhibit the desired smoothness when expressed as a linear filter. As suggested in prior studies [3, 49], a commonly adopted approach is to employ the ideal low-pass filter to enhance the presence of low-frequency components in the input interaction signals.

To formulate, the ideal s-pass filter is obtained from the SVD of the Gram matrix, and serves as a low-rank compression of an original matrix:

$$\tilde{G}_I = U \Sigma V^T \tag{27}$$

$$\mathcal{H}_s(G_I) = V_{:s} V_{:s}^T, \tag{28}$$

where the decomposed Gram matrix $\tilde{G}_I$“/> could be generally normalized to any order β and we default to $\beta = \frac{1}{2}$. The filter function for $\mathcal{H}_s$ is determined by the low-pass parameter s:

$$h(\lambda) = \begin{cases} 1 & \lambda \leq \lambda_s \\ 0 & otherwise. \end{cases} \tag{29}$$

The ideal low-pass filter enhances the input interaction signals by eliminating high-frequency noise and retaining the more informative low-frequency components.

4.2.4 Comprehensive Polynomial Filter. So far we have developed two key filters: the polynomial generalized Gram filter $\mathcal{H}_P$ and the ideal low-pass filter $\mathcal{H}_s$. The overall filtering process for the PolyCF can be expressed as follows:

$$\mathcal{H}_g r_u = (\mathcal{H}_P + \omega \mathcal{H}_s) r_u, \tag{30}$$

where the hyperparameter ω is for balancing the tradeoff between low-pass signal enhancement and polynomial approximation.

In practice, to reduce the computational complexity associated with obtaining the filtered signal in Equation (30), we adopt the factorized form of the comprehensive filters that fully leverage sparse matrix multiplication:

$$\begin{aligned} \mathcal{H}_g r_u &= \mathcal{H}_P r_u + \omega \mathcal{H}_s r_u \\ &= \frac{1}{|\Gamma|} \sum_{\gamma \in \Gamma} \sum_{k=0}^{K} \theta_k^{(\gamma)} P_k (D_I^{-\gamma} R^T D_U^{-1})(R D_I^{\gamma-1} r_u) \\ &\quad + \omega V_{:s} (V_{:s}^T r_u). \end{aligned} \tag{31}$$

4.3 Model Optimization

In the previous sections, we have outlined the filtering process of the polynomial Gram filter. Nonetheless, it still needs a target function to optimize the convolution kernel of $\mathcal{H}_P$. While there have been several widely applied loss functions for CF models, we propose a tailored objective for PolyCF that better aligns with its goal of optimally approximating the interaction graph filter.

4.3.1 Graph Optimization Objective. To optimize $\mathcal{H}_P$ as a low-pass filter, we adapt the graph filter optimization objective from prior works [56, 64]. For a model-predicted signal r_u^* and its corresponding ground truth signal r_u, the optimal filter minimizes the inner-product distance $\mathcal{L}_g = \langle r_u - r_u^*, r_u - r_u^* \rangle_g$ in a specific product space $\langle \cdot, \cdot \rangle_g$. In Euclidean space, this is equivalent to minimizing the Frobenius norm. To encourage smooth differences between r_u and r_u^*, we define the product space using the signal spectra, represented by the Laplacian quadratic form [50]:

$$\mathcal{L}_g = \frac{1}{2} ||\nabla (r_u - r_u^*)||_F^2 = \sum_{i,j} [\tilde{G}_I^{(\frac{1}{2})}]_{ij} (r_{u,i} - r_{u,j}^*)^2 = (r_u - r_u^*)^T (I - \tilde{G}_I^{(\frac{1}{2})})(r_u - r_u^*). \tag{32}$$

To optimize the graph filter for reconstructing interactions while maintaining invariance to permutations, the filtered r_u^* is obtained as:

$$r_u^* = \mathcal{H}_g (r_u + z), \; z \sim \mathcal{N}(0, \epsilon I), \tag{33}$$

where the hyperparameter ϵ controls the level of noise added to the original interaction signal of u.

The signal spectra defined in Equation (32) ensure the smooth differences between the predicted and the true signals, while reducing the overall scale of these differences. The objective $\mathcal{L}_g$ is introduced to prevent overly focusing on specific items when fitting interaction signals, which is particularly beneficial for dense interaction graphs where multiple non-zero items exist in the input signal. Moreover, by introducing random noise z to the input signal, the graph optimization objective in Equation (32) encourages the learned convolution kernel to behave as a robust filter against high-frequency noise in signals while preserving valuable low-frequency components.

4.3.2. Bayesian Ranking (BPR) Optimization. In addition to optimizing PolyCF as a personalized recommender model, we incorporate BPR [48] loss as a recommendation objective. The BPR objective function encourages the model to differentiate between positively and negatively interacted

items, formulated as follows:

$$\mathcal{L}_{bpr} = \sum_{(u,i,j)\in O} -\ln \sigma(r^*_{ui} - r^*_{uj}), \tag{34}$$

where $O = \{(u, i, j)|R_{u,i} = 1, R_{u,j} = 0\}$ denotes the set of sampled data pairs. During each iteration, negative items are randomly selected for each observed user–item interaction pair. While widely applied by many CF models [21, 26], the pairwise formulation of $\mathcal{L}_{bpr}$ ensures the model can effectively identify all positive interactions, even in cases where the interaction signal r_u is rather sparse. This property complements the $\mathcal{L}_g$ that focuses on the general difference between filtered and ground truth signals, providing a more comprehensive optimization strategy.

From the perspective of the optimization of the model, the convexity of the log-sigmoid function allows us to establish a lower bound for the BPR objective using Jensen's inequality:

$$\begin{aligned}\mathcal{L}_{bpr} &= \mathbb{E}_{(u,i,j)\in O, z\in\mathcal{N}(0,\epsilon I)}[-\ln \sigma(r^*_{ui} - r^*_{uj})] \\ &\geq \mathbb{E}_z\{-\ln \sigma(\mathbb{E}_{(u,i,j)}[r^*_{ui} - r^*_{uj}])\}.\end{aligned} \tag{35}$$

Given the lower bound of $\mathcal{L}_{bpr}$, we can further vectorize the expectation in the objective and rewrite the term with filtered graph signals:

$$\begin{aligned}\mathcal{L}_{bpr} &\geq -\frac{1}{n}\mathbb{E}_z[\ln \sigma(|r_u| r^{*T}_u \cdot r_u - (n - |r_u|) r^{*T}_u \cdot (\mathbf{1} - r_u))] \\ &= -\frac{|r_u|}{n}\mathbb{E}_z[\ln \sigma(r^{*T}_u \cdot (\frac{n - |r_u|}{|r_u|}(\mathbf{1} - r_u))] \\ &\approx -\mathbb{E}_z[\ln \sigma(r^{*T}_u \cdot (r_u - \mathbf{1}))].\end{aligned} \tag{36}$$

Where the approximation holds because of the fact that, in real-world scenarios, $n \gg |r_u|$. The lower bound in Equation (36) indicates that for PolyCF, the BPR objective can be interpreted as optimizing a specific logistic regression problem with the log-likelihood objective. Compared with $\mathcal{L}_g$ that indirectly enhances the model from a signal processing perspective, $\mathcal{L}_{bpr}$ directly optimizes PolyCF to better capture the characteristics of user interaction as binary distributions.

In summary, we combine the two target functions with a weight parameter α to formulate the general optimization objective of PolyCF:

$$\mathcal{L} = \mathcal{L}_g + \alpha\mathcal{L}_{bpr}. \tag{37}$$

4.4 Complexity Analysis

For an interaction system with m users and n items, the number of interaction is quantified by the count of non-zero values in the interaction matrix, denoted as

$$\#r = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\mathbf{1}_{R_{ij}\neq 0}. \tag{38}$$

In practice, the number of interactions $\#r$ is significantly smaller than the matrix size $m \times n$, indicating that R is a sparse matrix. Computing the filtered signal directly using Equation (30) may hinder the model from capitalizing on this sparsity. Therefore, we decompose the computation process into two parts in Equation (31) and analyze their computational costs separately to provide a more comprehensive complexity analysis.

Table 1. Computational Complexity of Preprocessing and Inference Stages, Respectively

Model	Preprocessing	Forwarding
MF	$O((s\#r + s^3)C)$	nd
LightGCN	$O(m + n + \#r)$	$O(K(m + n + \#r)d)$
CAGCN	$O((m + n)^3)$	$O(K(m + n + \#r)d)$
GF-CF	$O((s\#r + s^3)C + n^2)$	$O(n\#r)$
PolyCF	$O((s\#r + s^3)C + n^2)$	$O(K\|\Gamma\|n\#r)$

The first component in Equation (31), namely $\mathcal{H}_p r_u$, is obtained via a polynomial graph filter. We can represent the computation sequence as follows:

$$\frac{1}{|\Gamma|}\sum_{\gamma\in\Gamma}\{\sum_{k=0}^{K}[P_k(D_I^{-\gamma}R^T D_U^{-1})][R(D_I^{\gamma-1}r_u)]\}. \tag{39}$$

While the filter results from the polynomial basis P_k can be updated iteratively as the order increases, the computational complexity is given by:

$$O(K|\Gamma|(\#r + \#rn)) = O(K|\Gamma|n\#r). \tag{40}$$

The second component in Equation (31), i.e., $\mathcal{H}_s r_u$, can also be simplified via rearranging the computation sequence. The computational complexity can be represented with:

$$O(n \cdot s + n \cdot s) = O(n \cdot s). \tag{41}$$

The overall computational complexity in the inference stage of PolyCF is determined by:

$$O(K|\Gamma|n\#r + ns) = O(K|\Gamma|n\#r). \tag{42}$$

Given that the selected cut-off frequency $s < \#r$.

We summarize the computational complexity of PolyCF and several representative CF baselines in Table 1. The forward recommendation for one user is considered as the forwarding process for complexity computation. Here C is a constant related to the convergence setting in the SVD process when using power iteration-based solvers such as PROPACK [34].

4.5 Comparison with Existing Model

As a graph filter-based CF model, PolyCF is related to several existing methods that include:

— *GF-CF* [49]: GF-CF represents one of the pioneering works that introduced the integration of graph filters into CF models. Both GF-CF and PolyCF share a similar motivation, which is to overcome the limitations of embedding-based methods by utilizing item Gram graph filters. However, there are key distinctions between the two approaches. Specifically, GF-CF is built upon a fixed symmetrically normalized item Gram matrix as a linear response filter to input signals. In contrast, PolyCF utilizes multiple general normalizations of Gram filter $\tilde{G}^{(\gamma)}$ to fully exploit the comprehensive spectral features. Additionally, PolyCF extends the expressiveness of Gram filters by employing a series of polynomial basis functions and obtains a more flexible response pattern.

— *JGCF* [21]: JGCF introduces the use of the polynomial Jacobi graph convolution [56] to enhance the conventional graph convolution employed in LightGCN [26] and has achieved promising results. Nevertheless, JGCF's expressiveness upper limit remains constrained by the predefined node embedding size d. JGCF and PolyCF share common principles in employing

Table 2. Descriptive Statistics of the Used Datasets

Dataset	#User	#Item	#Interactions	Density
Amazon-Book	52,643	91,599	2,984,108	0.062%
Yelp2018	31,668	38,048	1,561,406	0.130%
Gowalla	29,858	40,981	1,027,370	0.084%
Electronics	517,255	124,123	4,669,514	0.007%

a more flexible and expressive polynomial graph operator by utilizing a variety of polynomial basis functions. However, as elaborated in Section 4.1, PolyCF offers more informative filter results due to its utilization of Gram filters, which provide a richer and more adaptable representation of the data. This distinction allows PolyCF to potentially outperform JGCF in terms of recommendation performance and expressive capacity.

—LinkProp [16]: LinkProp views CF as a community modeling and link prediction task, introducing four hyperparameters to normalize node degree matrices and enhance graph convolution. LinkProp-Multi further extends the forward process of LinkProp into multi-order propagation. From a GSP perspective, both PolyCF and LinkProp benefit from the expressiveness brought by customized convolution kernels. However, compared with the single graph filter of LinkProp methods, PolyCF formalizes this customization through a spectral framework, emphasizing the integration of filter outputs across multiple spectral spaces. Additionally, LinkProp's large search space, driven by its additional hyperparameters, makes manual fine-tuning computationally expensive. The model fine-tuning becomes extremely costly for LinkProp-Multi, which involves multiple propagation layers. In contrast, PolyCF adaptively identifies the optimal convolution kernel via learnable signal processing objectives, enhancing its potential of capturing subtle spectral information.

5 Experiment

In this section, we present a series of comprehensive experiments conducted on three CF datasets. These experiments aim to showcase the effectiveness and robustness of the proposed PolyCF. Additionally, we perform in-depth ablation and parameter studies to investigate the functionality of different modules within PolyCF and analyze its sensitivity to hyperparameters.

5.1 Experimental Settings

5.1.1 Dataset and Evaluation Metric. We evaluated the performance of PolyCF and compared baseline methods on three widely adopted datasets, namely *Amazon-Book*, *Yelp2018*, and *Gowalla*. We adopt the same train/test set split to keep consistency with previous works [26, 39, 49]. We randomly choose 10% of the train set to tune hyperparameters for our model.

In addition to the three widely used benchmarking datasets, we conduct further experiments to evaluate the effectiveness of PolyCF on scenarios involving extremely large-scale or sparse datasets. Specifically, we utilize a subset of the Amazon Reviews dataset [42], namely *Electronics*. We conduct the 5-core cleaning strategy [54] and retain only reviews with ratings of 4 or higher (out of 5). Compared with the other datasets, the *Electronics* subset is significantly larger and sparser, providing a robust benchmark to assess the model's performance in handling sparse data and its computational efficiency in large-scale interaction systems. The statistics of the used datasets are listed in Table 2.

To evaluate the performance of our PolyCF, we adopt two widely used evaluation metrics, namely Recall@K and NDCG@K. We set the value of K to 20 for consistency across different methods.

5.1.2 Compared Baselines. We compare the performance of the proposed PolyCF with a wide range of baseline models from three perspectives, listed as follows:

— *CF-Based Models*: Traditional collaborative filer methods that focus on depicting the collaborative affinity between users and items. Include models that are based on matrix factorization, i.e., NeuMF [28] and ENMF [6], which decompose and compress the original interaction, or based on two-tower structures, i.e., BPR [48], YoutubeDNN [8] and SimpleX [39], which aim at modeling the pairwise similarity between users and items.
— *GNN-Based Models*: Models that leverage GNNs to capture the high-order similarities between nodes on interaction graphs, including APPNP [17], LightGCN [26], DGCF [55], UltraGCN [40], CAGCN [59] and JGCF [21].
— *Graph Filter Models*: Models that integrate graph filters to processing interaction graph signals, including EASE^{R} [51], GF-CF [49], LinkProp-Multi [16], PGSP [36], and SGFCF [43].

5.1.3 Implementation Details. We have implemented the proposed PolyCF using the PyTorch framework. For the baseline methods we compare against, we either duplicate the performance results from their original papers or reproduce the results based on the open source implementations. In the case of our PolyCF, the normalization order set Γ consists of four values chosen from the interval $[-1, 1]$ with a fixed step size of 0.1. In other words, we tune the initial $\gamma_0 \in [-1, 0.6]$ and the Γ is parameterized as $\Gamma = \{\gamma_0 + 0.1i\}_{i=0}^{4}$. The cut-off frequency order s is tuned from $\{64, 128, 256, 512\}$. The polynomial order K is fixed at $K = 5$, since larger K would only bring marginal performance improvements according to our experiments. The weight parameter for the objective function is fixed at $\alpha = 1$. We apply a dropout method on the convolution kernel with dropout rates tuned from $\{0.2, 0.4, 0.6, 0.8\}$. The perturbation strength is fixed as $\epsilon = 1$. The convolution parameters are optimized using stochastic gradient descent with a learning rate of $lr = 10^{-3}$. Various polynomial basis functions can be used in Equation (25). We select P_k from several widely applied basis functions for signal filters including: Monomial basis, Chebyshev basis, Bernstein basis, Jacobi basis, and Hermite basis. We adopt the Jacobi basis as the default for PolyCF.

5.2 General Comparison

We conduct the general experiments of the proposed PolyCF and the compared methods on the aforementioned three datasets and record their recommendation performance. From the results reported in Table 3, we make the following observations:

— The proposed PolyCF outperforms currently advanced methods, achieving state-of-the-art recommendation performance across all datasets. Specifically, PolyCF outperforms the best baseline method by over 1.5%, 0.71%, and 0.94% relative improvement on Recall@20, over 2.9%, 0.68%, and 1.00% relative improvement on NDCG@20. This improvement underscores the effectiveness of the idea of integrating generalized polynomial Gram filters and ideal low-pass filters in recovering missing interactions from input signals.
— Graph-based methods exhibit superior performance compared to traditional CF methods, highlighting the advantages of leveraging graph structures to model collaborative affinity between users and items. Notably, graph filter-based methods consistently demonstrate superior and more stable performance in comparison to GNN-based models. This observation aligns with the insights derived from Theorem 4.3.
— Graph filter-based methods tend to excel on sparser datasets, such as Amazon-Book and Gowalla, while other embedding-based methods exhibit better performance on relatively denser datasets like Yelp2018. This pattern suggests that models can obtain significant benefits

Table 3. The Test Results of PolyCF and All Baseline Methods

	Method	Amazon-Book		Yelp2018		Gowalla	
		Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
CF-based models	BPR [48]	0.0250	0.0196	0.0433	0.0354	0.1291	0.1109
	NeuMF [28]	0.0258	0.0200	0.0451	0.0363	0.1399	0.1212
	ENMF [6]	0.0359	0.0281	0.0624	0.0515	0.1523	0.1315
	YoutubeDNN [8]	0.0502	0.0388	0.0686	0.0567	0.1754	0.1473
	SimpleX [39]	0.0583	0.0468	0.0701	0.0575	0.1872	0.1557
GNN-based models	APPNP [17]	0.0384	0.0299	0.0635	0.0521	0.1708	0.1462
	LightGCN [26]	0.0411	0.0315	0.0649	0.0530	0.1830	0.1554
	DGCF [55]	0.0422	0.0324	0.0654	0.0534	0.1842	0.1561
	UltraGCN [40]	0.0681	0.0556	0.0683	0.0561	0.1862	0.1580
	CAGCN* [59]	0.0510	0.0403	0.0708	0.0586	0.1878	0.1591
	JGCF [21]	0.0692	0.0559	0.0701	0.0579	0.1894	0.1593
Graph filter models	EASE^R [51]	0.0710	0.0567	0.0657	0.0552	0.1765	0.1467
	GF-CF [49]	0.0710	0.0584	0.0697	0.0571	0.1849	0.1518
	LinkProp-Multi [16]	0.0721	0.0588	0.0690	0.0571	0.1908	0.1573
	PGSP [36]	0.0712	0.0587	0.0710	0.0583	0.1916	0.1605
	SGFCF [43]	OOM	OOM	0.0682	0.0548	0.1773	0.1499
	PolyCF	**0.0730**[a]	**0.0605**[a]	**0.0715**[a]	**0.0590**[a]	**0.1934**[a]	**0.1621**[a]

The highest performance is emphasized with bold font and the second highest is marked with underlines.
[a]Indicates that PolyCF outperforms the best baseline model at a p-value < 0.05 level of paired t-test.

from the filter results that capture the global relationships within interaction graphs, particularly in cases where training data is sparse.

Additionally, we utilize Electronics subset and three sparse versions of Yelp2018 dataset, which retain random subsets of 5%, 15%, and 25% of the interactions from original training set. From the experimental results presented in Table 4, we can observe that graph filter methods achieve significant improvements over graph embedding methods on extremely large datasets. On the other hand, graph embedding methods show advantages when most of the interactions are taken from the interaction graph, thereby leading to low-connectivity graph filters. Notably, PolyCF outperforms all the baseline methods, demonstrating its flexibility when dealing with extremely large and sparse interaction systems, where the generalized Gram filter alleviates the sparsity issue with more comprehensive spectral information.

5.3 Ablation Study

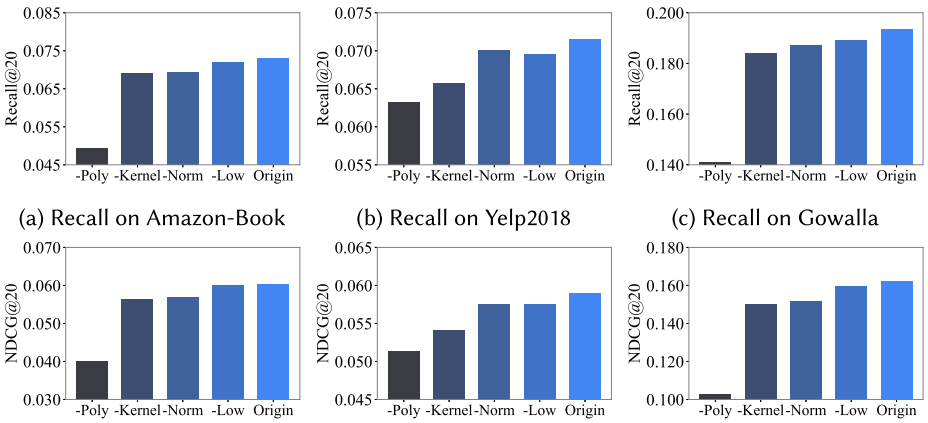
The proposed PolyCF incorporates a polynomial generalized Gram filter and an ideal low-pass filter to approximate the optimal filter function for recovering missing interactions. Additionally, a graph optimization objective function and a BPR ranking loss are leveraged to optimize the parameters of the polynomial filter. To gain a deeper understanding of the contributions of each module and how PolyCF benefits from the proposed optimization components, we conducted comprehensive ablation studies. These studies help dissect the role and impact of each element within PolyCF.

5.3.1 Functionality of Graph Filters. The two graph filters applied in PolyCF serve distinct purposes: the flexible polynomial generalized Gram filter aims to approximate the optimal response

Table 4. Performance Comparison of PolyCF and Several Representative Baseline Models on Extremely Large and Sparse Datasets

Method	Electronics		Yelp-5%		Yelp-15%		Yelp-25%	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
LightGCN [26]	0.0337	0.0246	0.0151	0.0126	0.0308	0.0252	0.0342	0.0287
JGCF [21]	0.0424	0.0318	0.0147	0.0116	0.0311	0.0239	0.0313	0.0265
GF-CF [49]	0.0694	0.0583	0.0113	0.0096	0.0210	0.0181	0.0295	0.0249
PGSP [36]	0.0724	0.0623	0.0125	0.0106	0.0271	0.0212	0.0300	0.0258
SGFCF [43]	OOM	OOM	0.0149	0.0137	0.0294	0.0241	0.0312	0.0259
PolyCF	0.0823	0.0691	0.0248	0.0198	0.0391	0.0304	0.0450	0.0346

Yelp-x% indicates the percentage of kept interactions from a training set.



(a) Recall on Amazon-Book (b) Recall on Yelp2018 (c) Recall on Gowalla

(d) NDCG on Amazon-Book (e) NDCG on Yelp2018 (f) NDCG on Gowalla

Fig. 3. Model performance with different settings.

function, while the ideal low-pass filter enhances low-pass components, thereby improving the model's expressiveness. To comprehensively assess the contributions of these components, we conducted experiments comparing the performance of PolyCF with four variants:

(1) W/O-Poly: This variant of PolyCF excludes the polynomial Gram filter, in other words, the input signals are solely processed with the ideal low-pass filter to make recommendation results.
(2) W/O-Kernel: In this variant, we replace the polynomial convolution kernel in PolyCF with a 1-order normalized Gram with fixed weight parameter.
(3) W/O-Norm: This variant of PolyCF removes all the other additional generally normalized Gram kernels, using only the polynomial of $\tilde{G}_I^{-\frac{1}{2}}$.
(4) W/O-Low: In this variant, we omit the ideal low-pass filter H_s and rely on the polynomial filter.

By comparing the performance of these variants, we can gain insights into the individual contributions of each module in PolyCF. Based on the results of ablation study presented in Figure 3, we can draw the following conclusions:

Table 5. Model Performance w.r.t. Optimization Targets

Method	Amazon-Book		Yelp2018		Gowalla		Electronics	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
GF-CF	0.0710	0.0584	0.0697	0.0571	0.1849	0.1518	0.0694	0.0583
LinkProp-Multi	0.0721	0.0588	0.0690	0.0571	0.1908	0.1573	0.0724	0.0623
PGSP	0.0712	0.0587	0.0710	0.0583	0.1916	0.1605	0.0732	0.0629
W/O-L_{bpr}	0.0718	0.0592	0.0709	0.0583	0.1920	0.1607	0.0593	0.0541
W/O-L_g	0.0721	0.0600	0.0712	0.0587	0.1912	0.1599	0.0820	0.0687
Origin	0.0730	0.0605	0.0715	0.0590	0.1934	0.1621	0.0823	0.0691

—All components of PolyCF help to improve model's capability of modeling the optimal graph filter. Specifically, when the polynomial filter is removed, PolyCF experiences the most significant performance decline, due to the removed relative signals. This observation indicates that the flexibility brought by a generalized polynomial graph filter is the key element for PolyCF to achieve its state-of-the-art performance, which illustrates the idea of extending handicraft graph filters into learnable graph convolution filters.
—Among the other components, the absence of a polynomial kernel in PolyCF leads to the most notable performance decline. This suggests that the presence of polynomial Gram filters empowers the model to effectively capture informative features and approximate the optimal filter function.

5.3.2 Effectiveness of Optimization Target. Recall that PolyCF is optimized via a pair of optimization targets: a graph-based optimization objective $\mathcal{L}_g$ and a Bayesian-based ranking loss $\mathbf{L}_{bpr}$. We conduct ablation studies on these two different optimization objectives to validate their effectiveness on PolyCF's performance. From the performance illustrated in Table 5, we can observe that:

—Both $\mathcal{L}_g$ and $\mathcal{L}_{bpr}$ are crucial to maintaining stable and adaptable performance across different datasets. This observation suggests that the two objective functions complement each other: the graph optimization objective formulates the model as a general, robust interaction filter, while the BPR loss drives PolyCF towards approximating the optimal filter for personalized rankings and recommendations.
—Specifically, the objective $\mathcal{L}_g$ has a greater impact on dense datasets (e.g. Gowalla), while $\mathcal{L}_{bpr}$ is the key for achieving superior performance on sparser datasets (Amazon-Books, Yelp2018, and Electronics). This observation aligns with the intuition behind the two objectives: the graph optimization excels in dense settings, while the pairwise ranking loss stabilizes performance in sparse scenarios.
—It is noteworthy that LinkProp-Multi outperforms several ablation baselines of PolyCF in Amazon-Book and Yelp2018 datasets, which implies the benefit of explicitly modeling the degree of influence. However, as discussed in previous sections, the performance of LinkProp model is highly dependent on finely tuned hyperparameters and lacks a comprehensive spectral perspective in its filtering process. These drawbacks limit its performance, particularly for datasets with large-scale or complex spectral characteristics, such as Electronics.

5.3.3 Compatibility of Generalized Normalization. As the proposed polynomial Gram filter can be integrated into other graph-based methods that leverage a normalized Laplacian of graphs, we validate the generalization capability of the filter by extending three representative Laplacian-based

Table 6. Performance of Applying Generalized Normalization Adjacency on Other Graph-Based Methods

Method	Amazon-Book		Yelp2018		Gowalla	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
LightGCN	0.0411	0.0315	0.0649	0.0530	0.1830	0.1554
LightGCN*	0.0431	0.0327	0.0654	0.0539	0.1856	0.1561
CAGCN	0.0516	0.0412	0.0699	0.0574	0.1884	0.1596
CAGCN*	0.0522	0.0419	0.0707	0.0581	0.1891	0.1598
JGCF	0.0692	0.0559	0.0701	0.0579	0.1894	0.1593
JGCF*	0.0699	0.0572	0.0704	0.0583	0.1898	0.1596
Origin	0.0730	0.0605	0.0715	0.0590	0.1934	0.1621

models: LightGCN, CAGCN (originally noted as CAGCN* in Table 3), and JGCF with the generalized polynomial filter. Specifically, *model** denotes the original *model* enhanced with a generalized kernel, with the hyperparameters for the convolution kernel specifically tuned for each model.

From the experimental results in Table 6, it is noteworthy that the application of our proposed generalized normalization form extends seamlessly to these graph Laplacian-based methods. The experimental results confirm its ability to capture rich spectral information from user interaction history. Additionally, given JGCF's inherent use of a polynomial graph convolution operator, incorporating our polynomial Gram filter aligns well with its existing structure, while the model could still benefit from the generalized normalization technique.

5.3.4 Influence of Chosen Polynomial Basis. The polynomial Gram filter in PolyCF is represented using a series of polynomial basis functions. Prior research on spectral GNNs [56] suggests that orthonormal basis functions in the polynomial space tend to offer optimal convergence performance. In other words, polynomial basis functions that satisfy the following condition:

$$I = \langle P_i, P_j \rangle = \int_{\lambda=0}^{2} P_i(\lambda) P_j(\lambda) W(\lambda) \mathrm{d}\lambda, \ \forall i, j \in \mathbb{N}^*, \tag{43}$$

where $W(\lambda)$ denotes a kernel space on which polynomial basis functions P_k are orthonormal to each other. Furthermore, one can construct new basis functions based on any arbitrary kernel metric $W(\lambda)$ using the Gram-Schmidt process. For instance, the Jacobian bases are constructed from the kernel metric of $W(\lambda) = (1-\lambda)^{\alpha}(1-\lambda)^{\beta}$ with α, β as predefined parameters.

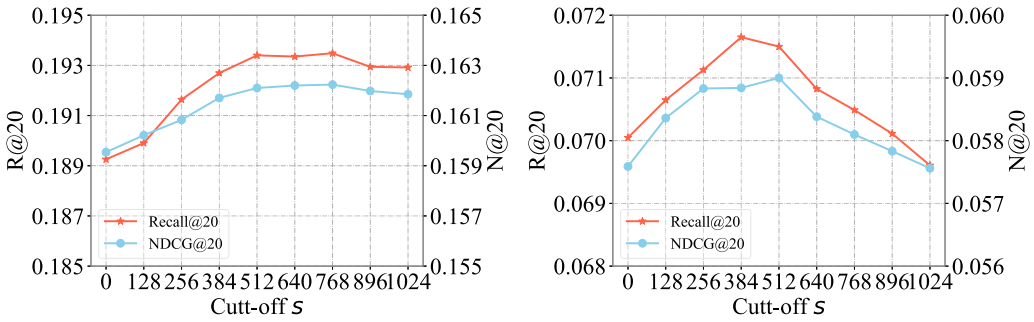
To assess the influence of the chosen polynomial basis, we evaluate the performance of PolyCF by replacing the polynomial basis P_k with several widely applied basis functions and the experimental results are presented in Table 7. The ablation study includes non-orthogonal bases including Monomial, Butterworth, and Berstein, as well as orthogonal bases including Hermitian, Chebyshev, and Jacobian bases. Notably, for the Chebyshev basis, we select Type I Chebyshev filters to better respond to the low-pass frequency signals.

Based on the experimental results, it is noteworthy that the choice of polynomial basis functions does not significantly impact the recommendation performance. Generally, PolyCF equipped with Jacobian filters as polynomial bases would achieve promising results; therefore, we choose Jacobian filters for other experiments by default.

Table 7. Model Performance w.r.t. Different Polynomial Basis of the Convolution Kernel

	Method	Amazon-Book		Yelp2018		Gowalla	
		Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
Non-orthogonal bases	Monomial	0.0531	0.0443	0.0605	0.0491	0.1741	0.1435
	Butterworth	0.0503	0.0420	0.0619	0.0502	0.1649	0.1355
	Berstein	0.0702	0.0571	0.0683	0.0561	0.1915	0.1608
Orthogonal bases	Hermite	**0.0732**	0.0602	0.0710	0.0584	0.1922	0.1611
	Chebyshev	0.0727	0.0604	0.0698	0.0579	**0.1936**	**0.1625**
	Jacobi	0.0730	**0.0605**	**0.0715**	**0.0590**	0.1934	0.1621

Basis with the best performance is highlighted in bold.



(a) Results on Gowalla

(b) Results on Yelp2018

Fig. 4. Model performance w.r.t. cut-off frequency s.

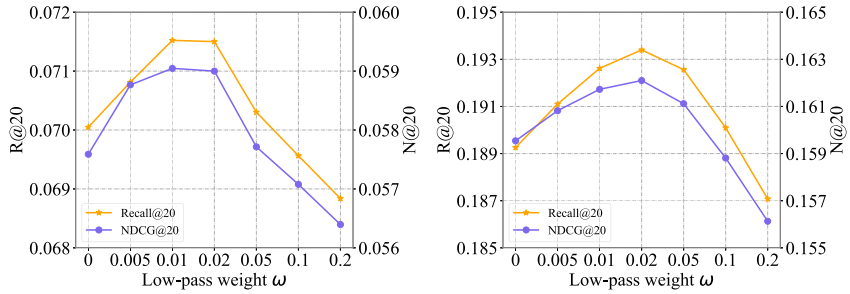
5.4 Parameter Analysis

The proposed PolyCF involves pre-defined hyperparameters that can significantly influence recommendation performance. Therefore, we conduct parameter studies to evaluate the sensitivity of PolyCF to these hyperparameters.

5.4.1 Influence of Cut-Off Frequency. The used ideal low-pass filter $\mathcal{H}_s$ depends on a cut-off frequency s that defines the bandwidth of the filtered components. We conducted a parameter study by adjusting the value of s and reporting the model's performance. From the results in Figure 4 we can draw the following conclusions:

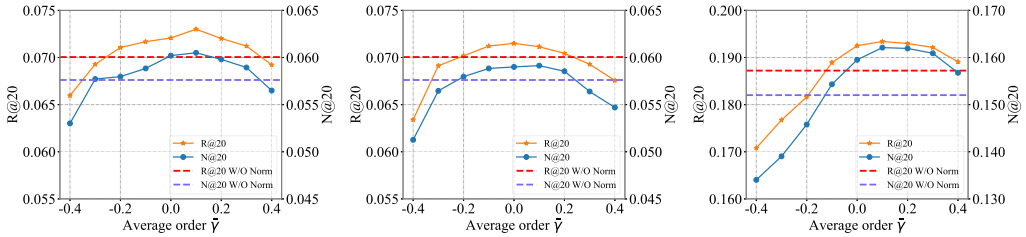
- —The choice of cut-off frequency s plays a crucial role in PolyCF's performance. A proper selection of s empowers PolyCF to effectively utilize the most informative part of each input interaction signal. In contrast, extremely large or small cut-off selection would lead to sub-optimal due to the missing components lying in the informative low-pass region of input signals.
- —The optimal cut-off frequency s varies across different datasets. This suggests that the most informative signal bandwidth for CF is flexible and can adapt to the characteristics of different user–item interactions.

5.4.2 Influence of Low-Pass Filter. The comprehensive graph filter of PolyCF is represented as a weighted sum of two separate filters. As shown in Equation (30), a weight parameter ω controls the low-pass components of input signals in model's recommendation results. To investigate the



(a) Performance on Yelp2018 (b) Performance on Gowalla

Fig. 5. PolyCF's performance w.r.t. low-pass weight ω.



(a) Results on Amazon-Books (b) Results on Yelp2018 (c) Results on Gowalla

Fig. 6. Model performance w.r.t. normalization orders Γ.

influence of ω, we conduct parameter studies and the model performance is shown in Figure 5. It can be observed that

- —The weight of low-pass components would significantly influence the overall model performance. While the model could suffer from non-significant low-pass signals, large amount of low-pass components would hinder the model from optimizing the kernel parameters.
- —While the optimal choice of the low-pass weight is data-specific, a relatively small ω would be beneficial for model performance in most cases.

5.4.3 Influence of Normalization Order Set. Recall that the generalized normalization order of PolyCF is selected from a parameter set Γ. The $\gamma \in \Gamma$ determines the model's perception of input signals from a spectral perspective, thereby affecting its recommendation performance. To verify the model's sensitivity to Γ, we conduct a parameter analysis and present the results in Figure 6. By comparing the model's performance with respect to the average normalization order $\bar{\gamma}$ and the baseline (which removes the generalized normalization), we can observe that the optimal choice of Γ varies across datasets. In general, a set of Γ values centered around positive values tends to maximize the benefits of multiple eigenspace perceptions. However, extreme values of γ may lead to performance degradation.

5.5 Convergence Efficiency of Models

In practice, the efficiency of model training and convergence is an important requirement for industry-level recommender systems. To access PolyCF's efficiency of convergence in real-world training scenarios, we record the time of training from the preprocessing phase until the optimal convergence epoch of PolyCF and several representative CF models. From the results presented in

Table 8. Convergence Time of PolyCF and Several Representative Baseline Methods

Model	Amazon-Book	Yelp2018	Gowalla	Electronics
LightGCN	77,304 s	54,417 s	32,829 s	362,063 s
CAGCN	38,267 s	14,160 s	10,611 s	180,177 s
JGCF	49,118 s	32,009 s	17,425 s	259,312 s
PolyCF	4,521 s	3,124 s	1,842 s	21,615 s

Table 9. Model Performance on Each User Group

Spectra (Top)	~20%	20~40%	40~60%	60~80%	80%~	Overall
W/O-Norm	0.1740↓↓	0.1748↓↓	0.2059↑	0.2058↑↑	0.1880−	0.1872
W/O-Low	0.1920↑	0.1848−	0.1921↑	0.1931↑	0.1621↓↓	0.1893
PolyCF	0.1867↓	0.1885−	0.2027↑	0.2090↑↑	0.1874↓	0.1934
Popularity (Top)	~20%	20~40%	40~60%	60~80%	80%~	Overall
W/O-Norm	0.1696↓↓	0.1816↓	0.1971↑	0.1944↑	0.2016↑↑	0.1872
W/O-Low	0.1740↓↓	0.1820↓	0.2051↑↑	0.1900−	0.1931↑	0.1893
PolyCF	0.1756↓↓	0.1853↓↓	0.2126↑↑	0.1932−	0.2117↑↑	0.1934

The marks ↑, ↓, and − indicate the users' group-aware performance against the overall model performance.

Table 8, we can observe PolyCF's remarkable advantage in convergence time as datasets scale up. This advantage attributes to its lower computational complexity per training epoch and relatively small parameter size. These characteristics of PolyCF enable it to convergence within a few epochs, comparing with embedding-based models that need to maintain the entire embedding table.

5.6 Interpretability of Recommendation Results

Interpretability is a crucial aspect of evaluating recommender models in real-world applications. To deepen our understanding, we conducted comprehensive studies on PolyCF's recommendation interpretability for diversified user groups. PolyCF benefits from the rich spectral information obtained by the polynomial Gram filter. To validate this, we categorized users based on their spectral and collaborative characteristics and evaluated PolyCF's recommendation performance on different user groups correspondingly.

To be specific, we assign each user u into clusters in two manners: from the spectral view, the spectra of r_u by calculating $r_u^T(I - \tilde{G}^{(\frac{1}{2})})r_u$; from the collaborative view, we obtain the degree of u on the user–item interaction graph $deg(u) = \sum_{i=0}^{n} R_{ui}$. Generally, r_u with higher spectra shows more oscillations in the spectral field, which reflects the user's diversified interests in different clusters of items. Meanwhile, users with higher degrees indicate his/her frequent interactions with a wide range of items. We equally split the users into five groups (from top 0% to top 100%) according to the two aforementioned indices and validate the model performance on each user group, respectively. The Recall@20 of recommendation results on different user groups on Gowalla dataset are listed in Table 9.

From the results, we can conclude the functionality of the generalize normalization of Gram matrix, as well as the idea low-pass filter of the PolyCF model. Specifically, the generalized Gram

Table 10. Model Performance on Perturbed Interactions

Model (Recall@20)	Missing (low)	Missing (high)	Noisy (low)	Noisy (high)	Popularity	Origin
LightGCN	0.1774	0.1752	-	-	0.1802	0.1830
PGSP	0.1847	0.1790	0.1835	0.1773	0.1884	0.1914
PolyCF	0.1911	0.1872	0.1914	0.1882	0.1901	0.1934
Model (NDCG@20)	Missing (low)	Missing (high)	Noisy (low)	Noisy (high)	Popularity	Origin
LightGCN	0.1460	0.1423	-	-	0.1513	0.1554
PGSP	0.1559	0.1463	0.1543	0.1431	0.1581	0.1604
PolyCF	0.1604	0.1573	0.1605	0.1580	0.1592	0.1621

filter provides rich spectral information, providing robustness when dealing with users with diversified spectral characteristics, as well as popular users that have wide ranges of interests. On the other hand, the ideal low-pass filter performs well on users with consistent interests and provides more constructive recommendations for users with less interaction history. Together, these modules enable PolyCF to offer comprehensive and interpretable recommendations.

5.7 Robustness of PolyCF

Being robust to different situations of inputs is important for real-world recommender systems. We conduct three sets of experiments to compare PolyCF's robustness with some widely used baselines. Specifically, we adopt three kinds of perturbed data, which are (1) missing interactions, where the observed interactions of r_u are randomly masked in a fixed ratio, (2) noisy data, where the input r_us are perturbed with a certain scale of Gaussian noises, and (3) popularity bias, where user's interactions with less popular items are more likely to be masked from r_u. By comparing the model performance under these three perturbation situations, we record the model performance on Gowalla datasets and the experimental results are listed in Table 10.

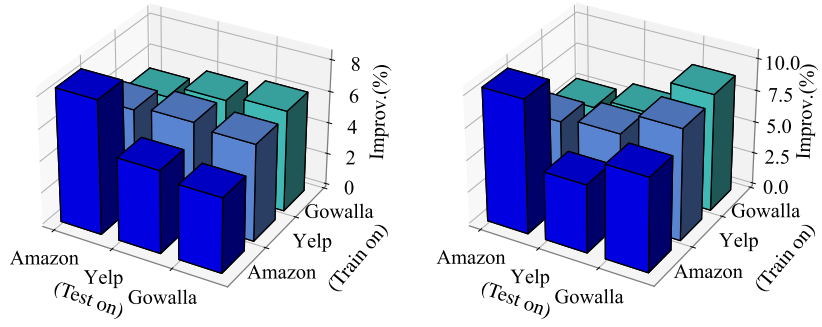
The experimental results demonstrate PolyCF's robustness compared to baseline models, maintaining resilience in the face of noisy inputs and popularity biases. This is because the graph optimization objective empowers PolyCF to filter out the high-pass noises as well as minor perturbations on the interaction signals.

5.8 In-Depth Study

5.8.1 Generalization of Convolution Kernel. The polynomial Gram filter is parameterized using a series of θ_i^γs to represent the convolution kernel. Previous research that employed straightforward, manually designed graph filters [49] has suggested the presence of shared similarities among the optimal response functions across different datasets. This raises the question of whether such generalization of simple filters still holds true for PolyCF.

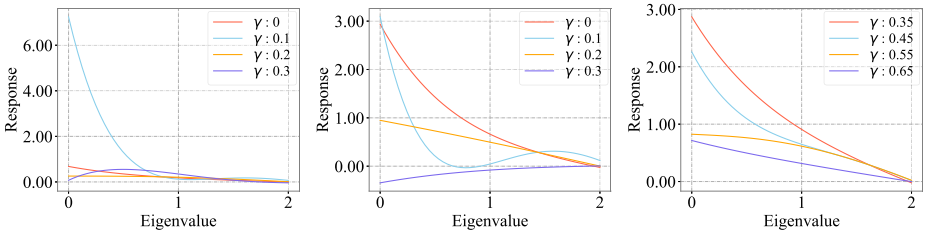
To investigate this, we test the generalization of filter parameters by firstly training PolyCF on three datasets to obtain the corresponding sets of θ values that characterize the optimal graph kernel. Subsequently, we apply these obtained sets of polynomial filters to other datasets to assess PolyCF's generalization capability. We report the relative improvement of the transferred results over the performance of PolyCF with randomly initialized parameters. From results in Figure 7 we can observe that, compared to randomly setting the convolution kernel, the parameters obtained from other datasets remain effective to a certain extent.

Additionally, it's noteworthy that the filter transferred from another dataset does not achieve comparable performance compared to the filter trained on the current dataset. This suggests that



(a) Recall@20 (b) NDCG@20

Fig. 7. PolyCF's generalization performance.



(a) Amazon-Book (b) Yelp2018 (c) Gowalla

Fig. 8. Visualization of PolyCF's convolution kernel.

there are distinctive features within interaction signals specific to different datasets. This observation also underscores the limitations of manually crafted filters, as they may struggle to approximate distinct optimal filter structures tailored to individual datasets.

5.8.2 Visualization of Polynomial Filter. The filtering process of the polynomial Gram filter depicts a response function that captures characteristic spectral features of input interaction signals. To intuitively show the learned response function, we visualize the response function and the corresponding polynomials based on the parameters of the convolution kernel obtained from the three datasets by combining the filter coefficients and corresponding polynomial basis.

From the visualization results in Figure 8, we can observe that the polynomial filter exhibits a more complex structure in its response function. This complexity arises from the distinct eigenspace structures introduced by the multiple generalized normalizations of the Gram matrix, enabling the polynomial convolution kernels to collaborate with each other and form a more expressive convolution structure. In general, the convolution kernel tends to resemble a low-pass filter, which aligns with empirical observations from prior research. However, there can still be a negative gain function corresponding to specific γs, which complements the general response function. More specifically, convolution kernels with smaller γ values tend to produce a steeper response function.

6 Conclusion

In this work, we dive into the expressiveness upper bound of previous node embedding-based CF methods. To break the limits of existing methods, we propose PolyCF, which is a graph filter-based model that solves the CF problem via GSP. Specifically, PolyCF incorporates a polynomial graph

filter, wherein the convolution kernel is constructed from a sequence of generalized normalization of Gram matrices. This design allows it to effectively capture distinctive spectral characteristics originating from diverse eigenspace structures. A graph optimization-based objective function as well as a BPR objective function are jointly used to optimize PolyCF, as an approximation of the optimal graph filter that can recover the missing interactions. An extensive set of experiments demonstrates that the proposed PolyCF attains state-of-the-art performance across three widely used CF datasets. Furthermore, our additional investigations verify the functionality of each component within PolyCF.

Acknowledgements

The authors are grateful to the anonymous reviewers for critically reading the manuscript and for giving important suggestions to improve the article.

References

[1] Sergey Brin. 1998. The PageRank citation ranking: Bringing order to the web. *Proceedings of ASIS 1998*, 98 (1998), 161–172.

[2] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems* 30, 1–7 (1998), 107–117.

[3] Xuheng Cai, Chao Huang, Lianghao Xia, and Xubin Ren. 2023. LightGCL: Simple yet effective graph contrastive learning for recommendation. arXiv:2302.08191. Retrieved from https://arxiv.org/abs/2302.08191

[4] Chao Chen, Haoyu Geng, Gang Zeng, Zhaobing Han, Hua Chai, Xiaokang Yang, and Junchi Yan. 2023. Graph signal sampling for inductive one-bit matrix completion: A closed-form solution. arXiv:2302.03933. Retrieved from https://arxiv.org/abs/2302.03933

[5] Chao Chen, Dongsheng Li, Junchi Yan, Hanchi Huang, and Xiaokang Yang. 2021. Scalable and explainable 1-bit matrix completion via graph signal learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, 7011–7019.

[6] Chong Chen, Min Zhang, Yongfeng Zhang, Yiqun Liu, and Shaoping Ma. 2020. Efficient neural matrix factorization without sampling for recommendation. *ACM Transactions on Information Systems* 38, 2 (2020), 1–28.

[7] Jeongwhan Choi, Seoyoung Hong, Noseong Park, and Sung-Bae Cho. 2023. Blurring-sharpening process models for collaborative filtering. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1096–1106.

[8] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for YouTube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, 191–198.

[9] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 3844–3852.

[10] Xiaowen Dong, Dorina Thanou, Laura Toni, Michael Bronstein, and Pascal Frossard. 2020. Graph signal processing for machine learning: A review and new perspectives. *IEEE Signal Processing Magazine* 37, 6 (2020), 117–127.

[11] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative memory network for recommendation systems. In *Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 515–524.

[12] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *Proceedings of the World Wide Web Conference*, 417–426.

[13] Wenqi Fan, Yao Ma, Dawei Yin, Jianping Wang, Jiliang Tang, and Qing Li. 2019. Deep social collaborative filtering. In *Proceedings of the 13th ACM Conference on Recommender Systems*, 305–313.

[14] Ziwei Fan, Ke Xu, Zhang Dong, Hao Peng, Jiawei Zhang, and Philip S. Yu. 2023. Graph collaborative signals denoising and augmentation for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2037–2041.

[15] Bin Feng, Zequn Liu, Nanlan Huang, Zhiping Xiao, Haomiao Zhang, Srbuhi Mirzoyan, Hanwen Xu, Jiaran Hao, Yinghui Xu, Ming Zhang, et al. 2024. A bioactivity foundation model using pairwise meta-learning. *Nature Machine Intelligence* 6, 8 (2024), 962–974.

[16] Hao-Ming Fu, Patrick Poirson, Kwot Sin Lee, and Chen Wang 2022. Revisiting neighborhood-based link prediction for collaborative filtering. In *Companion Proceedings of the Web Conference 2022*, 1009–1018.

[17] Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Predict then propagate: Graph neural networks meet personalized PageRank. arXiv:1810.05997. Retrieved from https://arxiv.org/abs/1810.05997

[18] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Communications of the ACM* 35, 12 (1992), 61–70.
[19] Prem Gopalan, Jake M. Hofman, and David M. Blei. 2015. Scalable recommendation with hierarchical Poisson factorization. In *Proceedings of the UAI*, 326–335.
[20] Marco Gori, Augusto Pucci, Via Roma, and I. Siena. 2007. Itemrank: A random-walk based scoring algorithm for recommender engines. In *Proceedings of the IJCAI*, Vol. 7, 2766–2771.
[21] Jiayan Guo, Lun Du, Xu Chen, Xiaojun Ma, Qiang Fu, Shi Han, Dongmei Zhang, and Yan Zhang. 2023. On manipulating signals of user-item graph: A Jacobi polynomial-based graph collaborative filtering. arXiv:2306.03624. Retrieved from https://arxiv.org/abs/2306.03624
[22] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 1025–1035.
[23] Bowen Hao, Hongzhi Yin, Jing Zhang, Cuiping Li, and Hong Chen. 2023. A multi-strategy-based pre-training method for cold-start recommendation. *ACM Transactions on Information Systems* 41, 2 (2023), 1–24.
[24] Mingguo He, Zhewei Wei, Zengfeng Huang, Hongteng Xu. 2021. BernNet: Learning arbitrary graph spectral filters via Bernstein approximation. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 14239–14251.
[25] Wei He, Guohao Sun, Jinhu Lu, and Xiu Susie Fang. 2023. Candidate-aware graph contrastive learning for recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1670–1679.
[26] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and powering graph convolution network for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 639–648.
[27] Xiangnan He, Ming Gao, Min-Yen Kan, and Dingxian Wang. 2016. Birank: Towards ranking on bipartite graphs. *IEEE Transactions on Knowledge and Data Engineering* 29, 1 (2016), 57–71.
[28] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, 173–182.
[29] Wei Ju, Zheng Fang, Yiyang Gu, Zequn Liu, Qingqing Long, Ziyue Qiao, Yifang Qin, Jianhao Shen, Fang Sun, Zhiping Xiao, et al. 2023. A comprehensive survey on deep graph representation learning. arXiv:2304.05055. Retrieved from https://arxiv.org/abs/2304.05055
[30] Wei Ju, Zhengyang Mao, Siyu Yi, Yifang Qin, Yiyang Gu, Zhiping Xiao, Yifan Wang, Xiao Luo, and Ming Zhang. 2024. Hypergraph-enhanced dual semi-supervised graph classification. arXiv:2405.04773. Retrieved from https://arxiv.org/abs/2405.04773
[31] Wei Ju, Yifang Qin, Siyu Yi, Zhengyang Mao, Kangjie Zheng, Luchen Liu, Xiao Luo, and Ming Zhang. 2023. Zero-shot node classification with graph contrastive embedding network. In *Transactions on Machine Learning Research*. Retrieved from https://openreview.net/forum?id=8wGXnjRLSy
[32] Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. arXiv:1609.02907. Retrieved from https://arxiv.org/abs/1609.02907
[33] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.
[34] R. M. Larsen. 2001. *Combining Implicit Restart and Partial Reorthogonalization in Lanczos Bidiagnalization*. Technical Report, SCCM, Stanford University.
[35] Fan Liu, Shuai Zhao, Zhiyong Cheng, Liqiang Nie, and Mohan Kankanhalli. 2024. Cluster-based graph collaborative filtering. *ACM Transactions on Information Systems* 42, 6 (2024), 1–24. DOI: https://doi.org/10.1145/3687481
[36] Jiahao Liu, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, Li Shang, and Ning Gu. 2023. Personalized graph signal processing for collaborative filtering. In *Proceedings of the ACM Web Conference 2023*, 1264–1272.
[37] Nian Liu, Xiao Wang, Deyu Bo, Chuan Shi, and Jian Pei. 2022. Revisiting graph contrastive learning from the perspective of graph spectrum. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2972–2983.
[38] Xiao Luo, Yusheng Zhao, Yifang Qin, Wei Ju, and Ming Zhang. 2023. Towards semi-supervised universal graph classification. *IEEE Transactions on Knowledge and Data Engineering* 36, 1 (2023), 416–428.
[39] Kelong Mao, Jieming Zhu, Jinpeng Wang, Quanyu Dai, Zhenhua Dong, Xi Xiao, and Xiuqiang He. 2021. SimpleX: A simple and strong baseline for collaborative filtering. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 1243–1252.
[40] Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. 2021. UltraGCN: Ultra simplification of graph convolutional networks for recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 1253–1262.

[41] Andriy Mnih and Russ R. Salakhutdinov. 2007. Probabilistic matrix factorization. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, 1257–1264.
[42] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP '19)*, 188–197.
[43] Shaowen Peng, Xin Liu, Kazunari Sugiyama, and Tsunenori Mine. 2024. How powerful is graph filtering for recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2388–2399.
[44] Yifang Qin, Wei Ju, Hongjun Wu, Xiao Luo, and Ming Zhang. 2023. Learning graph ODE for continuous-time sequential recommendation. arXiv:2304.07042. Retrieved from https://arxiv.org/abs/2304.07042
[45] Yifang Qin, Yifan Wang, Fang Sun, Wei Ju, Xuyang Hou, Zhe Wang, Jia Cheng, Jun Lei, and Ming Zhang. 2023. DisenPOI: Disentangling sequential and geographical influence for point-of-interest recommendation. In *Proceedings of the 16th ACM International Conference on Web Search and Data Mining*, 508–516.
[46] Ruihong Qiu, Hongzhi Yin, Zi Huang, and Tong Chen. 2020. Gag: Global attributed graph neural network for streaming session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 669–678.
[47] Raksha Ramakrishna, Hoi-To Wai, and Anna Scaglione. 2020. A user guide to low-pass graph signal processing and its applications: Tools and applications. *IEEE Signal Processing Magazine* 37, 6 (2020), 74–85.
[48] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. arXiv:1205.2618. Retrieved from https://arxiv.org/abs/1205.2618
[49] Yifei Shen, Yongji Wu, Yao Zhang, Caihua Shan, Jun Zhang, B. Khaled Letaief, and Dongsheng Li. 2021. How powerful is graph convolution for recommendation? In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 1619–1629.
[50] David I. Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine* 30, 3 (2013), 83–98.
[51] Harald Steck. 2019. Embarrassingly shallow autoencoders for sparse data. In *Proceedings of the World Wide Web Conference*, 3251–3257.
[52] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent relational metric learning via memory-based attention for collaborative ranking. In *Proceedings of the 2018 World Wide Web Conference*, 729–739.
[53] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-scale commodity embedding for e-commerce recommendation in Alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 839–848.
[54] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 165–174.
[55] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. 2020. Disentangled graph collaborative filtering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1001–1010.
[56] Xiyuan Wang and Muhan Zhang. 2022. How powerful are spectral graph neural networks. In *Proceedings of the International Conference on Machine Learning*. PMLR, 23341–23362.
[57] Yiqi Wang, Chaozhuo Li, Zheng Liu, Mingzheng Li, Jiliang Tang, Xing Xie, Lei Chen, and Philip S. Yu. 2022. An adaptive graph pre-training framework for localized collaborative filtering. *ACM Transactions on Information Systems* 41, 2 (Dec. 2022), Article 43, 27 pages. DOI: https://doi.org/10.1145/3555372
[58] Yifan Wang, Yifang Qin, Fang Sun, Bo Zhang, Xuyang Hou, Ke Hu, Jia Cheng, Jun Lei, and Ming Zhang. 2022. DisenCTR: Dynamic graph-based disentangled representation for click-through rate prediction. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2314–2318.
[59] Yu Wang, Yuying Zhao, Yi Zhang, and Tyler Derr. 2023. Collaboration-aware graph convolutional network for recommender systems. In *Proceedings of the ACM Web Conference 2023*, 91–101.
[60] Chunyu Wei, Jian Liang, Di Liu, and Fei Wang. 2022. Contrastive graph structure learning via information bottleneck for recommendation. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 20407–20420.
[61] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 726–735.
[62] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, 346–353.
[63] Jiafeng Xia, Dongsheng Li, Hansu Gu, Jiahao Liu, Tun Lu, and Ning Gu. 2022. FIRE: Fast incremental recommendation

with graph signal processing. In *Proceedings of the ACM Web Conference 2022*, 2360–2369.

[64] Keyulu Xu, Mozhi Zhang, Stefanie Jegelka, and Kenji Kawaguchi. 2021. Optimization of graph neural networks: Implicit acceleration by skip connections and more depth. In *Proceedings of the International Conference on Machine Learning*. PMLR, 11592–11602.

[65] Senrong Xu, Liangyue Li, Zenan Li, Yuan Yao, Feng Xu, Zulong Chen, Quan Lu, and Hanghang Tong. 2023. On the vulnerability of graph learning-based collaborative filtering. *ACM Transactions on Information Systems* 41, 4 (Mar. 2023), Article 87, 28 pages. DOI: https://doi.org/10.1145/3572834

[66] Junwei Yang, Hanwen Xu, Srbuhi Mirzoyan, Tong Chen, Zixuan Liu, Zequn Liu, Wei Ju, Luchen Liu, Zhiping Xiao, Ming Zhang, et al. 2024. Poisoning medical knowledge using large language models. *Nature Machine Intelligence* 6, 10 (2024), 1156–1168.

[67] Haibo Ye, Xinjie Li, Yuan Yao, and Hanghang Tong. 2023. Towards robust neural graph collaborative filtering via structure denoising and embedding perturbation. *ACM Transactions on Information Systems* 41, 3 (Feb. 2023), Article 59, 28 pages. DOI: https://doi.org/10.1145/3568396

[68] Junliang Yu, Xin Xia, Tong Chen, Lizhen Cui, Nguyen Quoc Viet Hung, and Hongzhi Yin. 2023. XSimGCL: Towards extremely simple graph contrastive learning for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 36, 2 (2023), 913–926.

[69] Ge Zhang, Zhao Li, Jiaming Huang, Jia Wu, Chuan Zhou, Jian Yang, and Jianliang Gao. 2022. eFraudCom: An E-commerce fraud detection system via competitive graph neural networks. *ACM Transactions on Information Systems* 40, 3 (Mar. 2022), Article 47, 29 pages. DOI: https://doi.org/10.1145/3474379

[70] Zhengbang Zhu, Rongjun Qin, Junjie Huang, Xinyi Dai, Yang Yu, Yong Yu, and Weinan Zhang. 2024. Understanding or manipulation: Rethinking online performance gains of modern recommender systems. *ACM Transactions on Information Systems* 42, 4 (Feb. 2024), Article 90, 32 pages. DOI: https://doi.org/10.1145/3637869

Received 19 August 2024; revised 22 February 2025; accepted 28 March 2025