Triple Structural Information Modeling for Accurate, Explainable and Interactive Recommendation

Jiahao Liu*
School of Computer Science
Fudan University
Shanghai, China
jiahaoliu21@m.fudan.edu.cn

Dongsheng Li
Microsoft Research Asia
Shanghai, China
dongsli@microsoft.com

Hansu Gu†
Seattle, United States
hansug@acm.org

Tun Lu*†
School of Computer Science
Fudan University
Shanghai, China
lutun@fudan.edu.cn

Peng Zhang*
School of Computer Science
Fudan University
Shanghai, China
zhangpeng_@fudan.edu.cn

Li Shang*
School of Computer Science
Fudan University
Shanghai, China
lishang@fudan.edu.cn

Ning Gu*
School of Computer Science
Fudan University
Shanghai, China
ninggu@fudan.edu.cn

ABSTRACT

In dynamic interaction graphs, user-item interactions usually follow heterogeneous patterns, represented by different structural information, such as user-item co-occurrence, sequential information of user interactions and the transition probabilities of item pairs. However, the existing methods cannot simultaneously leverage all three structural information, resulting in suboptimal performance. To this end, we propose TriSIM4Rec, a triple structural information modeling method for accurate, explainable and interactive recommendation on dynamic interaction graphs. Specifically, TriSIM4Rec consists of 1) a dynamic ideal low-pass graph filter to dynamically mine co-occurrence information in user-item interactions, which is implemented by incremental singular value decomposition (SVD); 2) a parameter-free attention module to capture sequential information of user interactions effectively and efficiently; and 3) an item transition matrix to store the transition probabilities of item pairs. Then, we fuse the predictions from the triple structural information sources to obtain the final recommendation results. By analyzing the relationship between the SVD-based and the recently emerging graph signal processing (GSP)-based collaborative filtering methods, we find that the essence of SVD is an ideal low-pass graph filter, so that the interest vector space in TriSIM4Rec can be extended to achieve explainable and interactive recommendation, making it possible for users to actively break through the information cocoons. Experiments on six public datasets demonstrated the effectiveness of TriSIM4Rec in accuracy, explainability and interactivity.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

recommendation system, singular value decomposition, graph filtering, user behavior modeling

ACM Reference Format:
Jiahao Liu, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, Li Shang, and Ning Gu. 2023. Triple Structural Information Modeling for Accurate, Explainable and Interactive Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3539618.3591779

1 INTRODUCTION

Real-time modeling and prediction of user interactions are widely used in recommender systems [13–15, 31]. In most cases, we can only observe the interaction data between users and items, which may occur due to heterogeneous patterns according to the characteristics of the applications. There are three types of key structural information from the heterogeneous patterns: (1) *co-occurrence information* contained in user-item interaction graph, (2) *sequential information* of user interactions and (3) *item transition information* between item pairs. However, the existing methods do not make full use of these three types of structural information. Sequential methods [18, 33] model users as a sequence of items, without explicitly modeling the rich co-occurrence information between users and

*Also with Shanghai Key Laboratory of Data Science, Fudan University, China, and Shanghai Institute of Intelligent Electronics & Systems, China.
†Corresponding author.

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07…$15.00
https://doi.org/10.1145/3539618.3591779

items. Graph-based methods [23, 41] directly model users on interaction graphs, but cannot model sequential information and item transition information containing rich personalized information.

In this paper, we propose TriSIM4Rec — a triple structural information modeling method for recommendation, which can effectively leverage all three kinds of structural information simultaneously to achieve accurate, explainable and interactive recommendation. Specifically, we use singular value decomposition (SVD) to mine co-occurrence information on the user-item interaction graph, use a parameter-free attention mechanism to capture sequential information in user interaction sequence, and construct item transition matrix to store item transition information. Moreover, we use incremental SVD [2] to update TriSIM4Rec incrementally, and model user interaction behavior in a parameter-free manner, which makes TriSIM4Rec very efficient.

While providing convenience for users to access items, recommender systems may also place users in the information cocoons [30], in which the users' interactions could be significantly affected by the exposure bias of the recommender systems [6], i.e., users passively select from the recommended items without having sufficient freedom of exploring their diverse interests. To this end, we extend TriSIM4Rec to an explainable and interactive recommendation method by building an understandable latent space and enabling controllable recommendations based on this latent space.

To achieve this, we first understand SVD from the perspective of graph signal processing (GSP), and show that truncated SVD [9] is equivalent to a low-pass graph filter. This means that the incremental SVD used in TriSIM4Rec is essentially a *dynamic low-pass graph filter*, which mines co-occurrence information in user-item interactions in a dynamic manner. Specifically, we first show that the collaborative filtering based on graph signal processing can be implemented by SVD. Then, we understand SVD from the perspective of graph filtering, and show that SVD maps users and items to a Fourier space defined by user similarity graph and a Fourier space defined by item similarity graph respectively, and finally maps users and items to the same Fourier space through scaling transformation.

Following the above understanding, the decomposition of the user-item interaction matrix using SVD can map users from the *item vector space* to the *interest vector space*, where the item vector space is a concept of spatial domain and the interest vector space is a concept of the spectral domain from the perspective of graph filtering. In the spectral domain, by controlling the proportion of signals with different frequencies, users can control the proportion of items representing different interests in the recommendation results, and then customize their own recommendation results.

To analyze the performance of TriSIM4Rec, we conduct detailed experiments on two recommendation tasks (future item recommendation and next interaction prediction), which show that TriSIM4Rec can substantially outperform the state-of-the-art methods in accuracy while achieving high computation efficiency and high robustness on very sparse data. Our ablation studies also confirm that all three kinds of structural information contribute to the performance improvement of TriSIM4Rec. Moreover, we visualize the explanations and interact with the recommendation models through case studies, and the results show that TriSIM4Rec can achieve satisfactory explainability and interactivity.

The main contributions of this paper are summarized as follows:

- We propose TriSIM4Rec, an effective and efficient recommendation method, which can improve the accuracy by leveraging the three types of structural information simultaneously and improve the efficiency by incrementally updating a parameter-free model.
- We understand SVD from the perspective of graph signal processing and show that truncated SVD is equivalent to a low-pass graph filter. This connection enables the understanding of user interests in the spectral domain.
- We propose the concepts of item vector space in the spatial domain and interest vector space in the spectral domain, and extend TriSIM4Rec to an explainable and interactive method, so that users can actively break through the information cocoons by interacting with the recommendation model.

2 RELATED WORK

In this section, we introduce the work related to dynamic user behavior modeling.

Sequential methods. One line of works regards the occurrence of interaction events between users and items as a temporal point process, and models the interactions through the intensity functions [33, 44]. Wang et al. [35] model the co-evolving nature of users and items through a co-evolutionary process. Shchur et al. [26] directly model the conditional distribution of inter-event times. Cao et al. [3] incorporate topology and long-term dependencies into the intensity function. The other line of works is based on the recurrent neural network (RNN) [1, 5, 7, 18, 37, 43], which usually uses coupled RNNs to model users and items respectively. For instance, DeePRed [16] employs non-recursive mutual RNNs to model interactions.

Graph-based methods. Graph-based methods [21–23, 41] can directly model users and items on the interaction graphs. TDIG-MPNN [4] models global and local information simultaneously. DGCF [20] updates users and items through three mechanisms. SDGNN [32] takes the state changes of neighbors into account. MetaDyGNN [39] proposes a meta-learning framework for few-shot link prediction. TREND [36] proposes a Hawkes process-based graph neural network (GNN). FIRE [38] proposes a temporary information filter to dynamically model users and items. IGE [40] generates embeddings with two coupled networks, and TigeCMN [42] further incorporates memory networks.

3 METHOD

In this section, we first introduce the architecture of TriSIM4Rec, and then introduce how TriSIM4Rec achieves incremental updates.

3.1 Architecture

The architecture of TriSIM4Rec is shown in Figure 1. By mining the co-occurrence information, sequential information and item transition information in the user-item interactions, the *co-occurrence score*, *sequential score* and *transition score* are obtained respectively, and finally fused into the *final score*.

Problem description. Let user set be $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ and item set be $\mathcal{I} = \{i_1, i_2, ..., i_{|\mathcal{I}|}\}$, where $|\cdot|$ is the number of elements

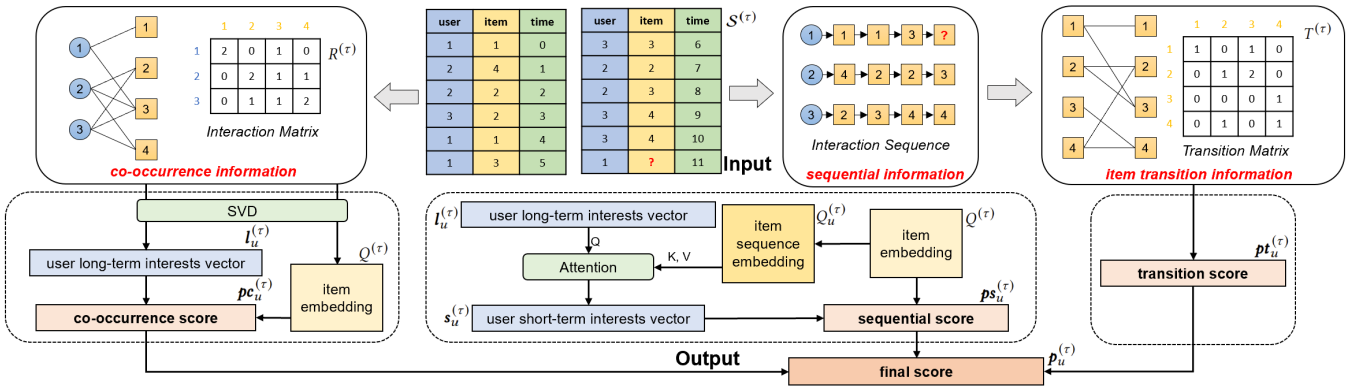

Figure 1: The architecture of TriSIM4Rec.

in a set. Without loss of generality, we use u to represent a user and i to represent an item when their indices are not concerned. Each user-item interaction can be represented by a 3-tuple (u, i, t), where t is the timestamp of the interaction. When there are τ interactions in total, we can represent them as a sequence $\mathcal{S}^{(\tau)} =< (u^{(1)}, i^{(1)}, t^{(1)}), (u^{(2)}, i^{(2)}, t^{(2)}), ..., (u^{(\tau)}, i^{(\tau)}, t^{(\tau)}) >$. Now given a user u, we need to predict which item that user u will interact with in the $\tau + 1$-th interaction. The output of the model is an $|\mathcal{I}|$-dimensional vector, and each dimension represents the possibility of interaction between the user and the corresponding item.

Next, we will introduce how to use these three types of structural information and how we get the final score in detail.

3.1.1 Co-occurrence Information. The co-occurrence information means that users with similar interaction history will interact with similar items in the future, which is the basic idea of collaborative filtering [8, 17, 19, 25]. We construct a new user-item *interaction matrix* to model co-occurrence information. When constructing the interaction matrix, we introduce time decay to measure the interaction score between users and items, so that the model can focus more on the recent interactions. Let $\tilde{R}^{(\tau)} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ be the interaction matrix at $t^{(\tau)}$, then row u and column i of $\tilde{R}^{(\tau)}$ is:

$$\tilde{R}^{(\tau)}[u, i] = \sum_{(u,i,t)\in\mathcal{S}^{(\tau)}} \sigma_t(t), \tag{1}$$

where $\sigma_t(t) = exp\{-\beta_t(1 - t/t^{(\tau)})\}$ is the time decay function, and β_t is time decay coefficient. We normalize $\tilde{R}^{(\tau)}$ to mitigate the popularity deviation [28, 29], and get the final interaction matrix at $t^{(\tau)}$:

$$R^{(\tau)} = diag(\boldsymbol{d}_U^{(\tau)})^{-1/2}\tilde{R}^{(\tau)}diag(\boldsymbol{d}_I^{(\tau)})^{-1/2}, \tag{2}$$

where $diag(\cdot)$ represents a diagonal matrix. The u-th element of $\boldsymbol{d}_U^{(\tau)} \in \mathbb{R}^{|\mathcal{U}|}$ and the i-th element of $\boldsymbol{d}_I^{(\tau)} \in \mathbb{R}^{|\mathcal{I}|}$ are:

$$\boldsymbol{d}_U^{(\tau)}[u] = \sum_{j=1}^{|\mathcal{I}|}\tilde{R}^{(\tau)}[u, j], \quad \boldsymbol{d}_I^{(\tau)}[i] = \sum_{j=1}^{|\mathcal{U}|}\tilde{R}^{(\tau)}[j, i]. \tag{3}$$

Truncated SVD [24] can mine co-occurrence information for the essence of truncated SVD is a low-pass graph filter, which will be analyzed in Section 4. We obtain the low-rank approximation of $R^{(\tau)}$ through truncated SVD: $R^{(\tau)} \approx U^{(\tau)}\Sigma^{(\tau)}(V^{(\tau)})^\top$, where

$$\begin{aligned} U^{(\tau)} &= (\boldsymbol{u}_1^{(\tau)}, \boldsymbol{u}_2^{(\tau)}, ..., \boldsymbol{u}_k^{(\tau)}) \in \mathbb{R}^{|\mathcal{U}|\times k}, \\ \Sigma^{(\tau)} &= diag((s_1^{(\tau)}, s_2^{(\tau)}, ..., s_k^{(\tau)})) \in \mathbb{R}^{k\times k}, s_1^{(\tau)} > ... > s_k^{(\tau)}, \\ V^{(\tau)} &= (\boldsymbol{v}_1^{(\tau)}, \boldsymbol{v}_2^{(\tau)}, ..., \boldsymbol{v}_k^{(\tau)}) \in \mathbb{R}^{|\mathcal{I}|\times k}. \end{aligned} \tag{4}$$

$s_j^{(\tau)}$ is the square root of the j-th largest eigenvalue of $R^{(\tau)}(R^{(\tau)})^\top$ or $(R^{(\tau)})^\top R^{(\tau)}$, $\boldsymbol{u}_j^{(\tau)}$ and $\boldsymbol{v}_j^{(\tau)}$ are the eigenvectors of $R^{(\tau)}(R^{(\tau)})^\top$ and $(R^{(\tau)})^\top R^{(\tau)}$ corresponding to $s_j^{(\tau)}$, respectively, $(j = 1, 2, ..., k)$. Then, we can obtain *user embedding* $P^{(\tau)}$ and *item embedding* $Q^{(\tau)}$ at $t^{(\tau)}$:

$$P^{(\tau)} = U^{(\tau)}(\Sigma^{(\tau)})^{1/2}, \quad Q^{(\tau)} = V^{(\tau)}(\Sigma^{(\tau)})^{1/2}. \tag{5}$$

We will see that $P^{(\tau)}$ will change smoothly with new interactions occurring in Section 3.2, which is consistent with a user's long-term interests reflected by the interaction matrix. Thus, for a user u, we define the u-th row of $P^{(\tau)}$ as the *user long-term interests vector*, which is formally described as follows:

$$\boldsymbol{l}_u^{(\tau)} = P^{(\tau)}[u, \cdot] \in \mathbb{R}^{1\times k}. \tag{6}$$

Similarly, $Q^{(\tau)}$ also changes smoothly with new interactions occurring, which reflects the co-evolving nature of users and items [7, 35].

The co-occurrence scores of user u on all items at time $t^{(\tau)}$ can be obtained as follows:

$$\boldsymbol{pc}_u^{(\tau)} = \boldsymbol{l}_u^{(\tau)}(Q^{(\tau)})^\top \in \mathbb{R}^{|\mathcal{I}|}. \tag{7}$$

3.1.2 Sequential Information. User interaction sequence contains rich personalized information, so we model sequential information through attention mechanism [34]. The interaction sequence of user u at $t^{(\tau)}$ is $\mathcal{S}_u^{(\tau)} =< (i_u^{(1)}, t_u^{(1)}), (i_u^{(2)}, t_u^{(2)}), ..., (i_u^{(n_u^\tau)}, t_u^{(n_u^\tau)}) >$, where $i_u^{(j)}$ is the item id of the j-th interaction of user u, $t_u^{(j)}$ is the timestamps of the j-th interaction of user u, n_u^τ is the number of interactions of user u at time $t^{(\tau)}$. For a user u, we get her/his *item sequence embedding* matrix by arranging the embeddings of items she/he has interacted with in rows after time decay, represented by $Q_u^{(\tau)} \in \mathbb{R}^{n_u^\tau\times k}$. Time decay plays the role of position embedding, making the model pay more attention to recent interactions and

focus on short-term interests. We model the short-term interests of user u as follows:

$$\boldsymbol{s}_u^{(\tau)} = softmax(\frac{\boldsymbol{l}_u^{(\tau)}(Q_u^{(\tau)})^\top}{\sqrt{k}})Q_u^{(\tau)} \in \mathbb{R}^{1\times k}. \tag{8}$$

Unlike the user long-term interests vector, $\boldsymbol{s}_u^{(\tau)}$ is a linear combination of items that a user has interacted with, which will change dramatically when new interactions occur, so we call $\boldsymbol{s}_u^{(\tau)}$ *user short-term interests vector*. The weight of the linear combination is related to the inner product of $\boldsymbol{l}_u^{(\tau)}$ and each row of $Q_u^{(\tau)}$, which means that the more similar an item is to a user's long-term interests, the more it is to describe the user's short-term interests. It should be noted that the $\boldsymbol{l}_u^{(\tau)}$ and $Q_u^{(\tau)}$ are obtained through SVD and are in the same embedding space, which will be detailed in Section 4. Therefore, there is no need for feature transformation, so there are no learnable parameters in Eq. (8).

The sequential scores of user u on all items at time $t^{(\tau)}$ can be obtained as follows:

$$\boldsymbol{ps}_u^{(\tau)} = \boldsymbol{s}_u^{(\tau)}(Q^{(\tau)})^\top \in \mathbb{R}^{|\mathcal{I}|}. \tag{9}$$

3.1.3 Item transition Information. The item transition information is not user-specific but statistical, reflecting the overall preference of users. We construct a *transition matrix* to model the item transition information. For any user $u \in \mathcal{U}$, we define the *transition* from item $i_u^{(j)}$ to item $i_u^{(j+1)}$ in $\mathcal{S}_u^{(\tau)}$ as $(i_u^{(j)}, t_u^{(j)}) \xrightarrow{\mathcal{S}_u^{(\tau)}} (i_u^{(j+1)}, t_u^{(j+1)})$, $j = 1, 2, ..., n_u^\tau - 1$. The element at the i_1-th row and i_2-th column of the transition matrix $T^{(\tau)} \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{I}|}$ at time $t^{(\tau)}$ is

$$T^{(\tau)}[i_1, i_2] = \sum_{u\in\mathcal{U}} \sum_{(i_1,t_1)\xrightarrow{\mathcal{S}_u^{(\tau)}}(i_2,t_2)} \sigma_i(t_1, t_2), \tag{10}$$

where $\sigma_i(t_1, t_2) = exp\{\beta_i(t_2 - t_1)/t^{(\tau)}\}$ is the interval decay function, and β_i is interval decay coefficient.

For a user u, we use the row corresponding to the item that the user most recently interacted with in the transition matrix as the transition score:

$$\boldsymbol{pt}_u^{(\tau)} = T^{(\tau)}[i_u^{(n_u^{(\tau)})}, \cdot] \in \mathbb{R}^{|\mathcal{I}|}. \tag{11}$$

3.1.4 Fusion for Final Score. The final score can be obtained by the weighted sum of the three scores described earlier:

$$\boldsymbol{p}_u^{(\tau)} = (1-\lambda_t)((1-\lambda_s)\boldsymbol{pc}_u^{(\tau)} + \lambda_s\boldsymbol{ps}_u^{(\tau)}) + \lambda_t\boldsymbol{pt}_u^{(\tau)} \in \mathbb{R}^{|\mathcal{I}|}, \tag{12}$$

where λ_s is the short-term interests coefficient that controls the ratio between short-term interests (sequential score) and long-term interests (co-occurrence score), λ_t is the item transition information coefficient that controls the weight of the transition score. Each element of $\boldsymbol{p}_u^{(\tau)}$ represents the prediction score of interaction between user u and the corresponding item at $t^{(\tau+1)}$.

3.2 Incremental Update

User interests usually change over time, which are reflected by continuous interactions. Therefore, TriSIM4Rec needs to be updated in real-time according to the recent interactions to capture the latest interests of the users.

Problem Description. For any user u, assume that we have obtained $\boldsymbol{p}_u^{(\tau)}$, which will be used to predict which item that user u will interact with in the τ+1-th interaction. Then, we have observed the τ + 1-th interaction event $(u^{(\tau+1)}, i^{(\tau+1)}, t^{(\tau+1)})$. For the convenience of description, we use u and i to refer to $u^{(\tau+1)}$ and $i^{(\tau+1)}$ respectively, i.e., the new interaction is $(u, i, t^{(\tau+1)})$, and the last item that user u interact with before $t^{(\tau+1)}$ is i_0 with the interaction timestamp t_0. The task is to incrementally calculate $\boldsymbol{p}_u^{(\tau+1)}$.

In Eq. (12), the output $\boldsymbol{p}_u^{(\tau+1)}$ is the weighted sum of co-occurrence score $\boldsymbol{pc}_u^{(\tau+1)}$, sequential score $\boldsymbol{ps}_u^{(\tau+1)}$, and transition score $\boldsymbol{pt}_u^{(\tau+1)}$. Next, we will introduce how these three scores are obtained.

3.2.1 The Update of Co-occurrence Score. In Eq. (6) and (7), the update of $\boldsymbol{pc}_u^{(\tau+1)}$ depends on $\boldsymbol{l}_u^{(\tau+1)}$ and $Q^{(\tau+1)}$, and $\boldsymbol{l}_u^{(\tau+1)}$ depends on $P^{(\tau+1)}$. In Eq (4) and (5), the calculations of $P^{(\tau+1)}$ and $Q^{(\tau+1)}$ rely on truncated SVD to obtain $U^{(\tau+1)}$, $\Sigma^{(\tau+1)}$, and $V^{(\tau+1)}$ first, which is very time-consuming. Therefore, if we want to get $\boldsymbol{pc}_u^{(\tau+1)}$ in real-time, we need a more efficient update algorithm to obtain $U^{(\tau+1)}$, $\Sigma^{(\tau+1)}$, and $V^{(\tau+1)}$.

For any matrix $A^0 \in \mathbb{R}^{m\times n}$, if we have decomposed it by truncated SVD, and get U_A^0, Σ_A^0, and V_A^0 such that $A^0 \approx U_A^0\Sigma_A^0(V_A^0)^\top$. Then A^0 gets an increment matrix $\Delta A = \boldsymbol{ab}^\top$ and becomes $A^1 = A^0 + \Delta A$, where $\boldsymbol{a} \in \mathbb{R}^m$ and $\boldsymbol{b} \in \mathbb{R}^n$ are both column vectors. As shown in Algorithm 1, incremental SVD [2] provides a fast calculation method to get U_A^1, Σ_A^1 and V_A^1 incrementally, in which $A^1 \approx U_A^1\Sigma_A^1(V_A^1)^\top$. We denote this update process by $U_A^1, \Sigma_A^1, V_A^1 \leftarrow iSVD(U_A^0, \Sigma_A^0, V_A^0, \boldsymbol{a}, \boldsymbol{b})$.

However, the increment matrix of interaction matrix $\Delta R^{(\tau)} = R^{(\tau+1)} - R^{(\tau)}$ is a rank-2 matrix, which means that $\Delta R^{(\tau)}$ cannot be expressed in the form of the product of two vectors. Therefore, $U^{(\tau+1)}$, $\Sigma^{(\tau+1)}$, and $V^{(\tau+1)}$ cannot be obtained directly through incremental SVD. As shown in Algorithm 2, to solve this problem, we disassemble $\Delta R^{(\tau)}$ into three rank-1 matrices as follows:

$$\Delta R^{(\tau)} = \boldsymbol{e}_u^{|\mathcal{U}|}(\boldsymbol{\Delta}_u)^\top + \boldsymbol{\Delta}_i(\boldsymbol{e}_i^{|\mathcal{I}|})^\top + \Delta_{ui}\cdot\boldsymbol{e}_u^{|\mathcal{U}|}(\boldsymbol{e}_i^{|\mathcal{I}|})^\top, \tag{13}$$

where $\boldsymbol{e}_j^n$ is a n-dimensional column vector, with only the j-th dimension being 1 and the rest being 0, and $\boldsymbol{\Delta}_u$, $\boldsymbol{\Delta}_i$, and Δ_{ui} are obtained by line 3, 4, and 8 of Algorithm 2, respectively. Then we execute the incremental SVD algorithm three times to get $U^{(\tau+1)}$, $\Sigma^{(\tau+1)}$, and $V^{(\tau+1)}$.

Then, we can get $P^{(\tau+1)}$, $Q^{(\tau+1)}$ and $\boldsymbol{l}_u^{(\tau+1)}$ by following Eq. (5) and (6), and get the updated co-occurrence score at $t^{(\tau+1)}$ as:

$$\boldsymbol{pc}_u^{(\tau+1)} = \boldsymbol{l}_u^{(\tau+1)}(Q^{(\tau+1)})^\top. \tag{14}$$

Algorithm 1: Incremental SVD [2]

Input: $U_A^0, \Sigma_A^0, V_A^0, \boldsymbol{a}, \boldsymbol{b}$
Output: U_A^1, Σ_A^1, V_A^1
1 $\boldsymbol{m} \leftarrow (U_A^0)^\top\boldsymbol{a}, \boldsymbol{p} \leftarrow \boldsymbol{a} - U_A^0\boldsymbol{m}, P \leftarrow ||\boldsymbol{p}||^{-1}\boldsymbol{p}$;
2 $\boldsymbol{n} \leftarrow (V_A^0)^\top\boldsymbol{b}, \boldsymbol{q} \leftarrow \boldsymbol{b} - V_A^0\boldsymbol{n}, Q \leftarrow ||\boldsymbol{q}||^{-1}\boldsymbol{q}$;
3 $K \leftarrow \begin{bmatrix}\Sigma_A^0 & \boldsymbol{0}\\ \boldsymbol{0} & 0\end{bmatrix} + \begin{bmatrix}\boldsymbol{m}\\ ||\boldsymbol{p}||\end{bmatrix}\begin{bmatrix}\boldsymbol{n}\\ ||\boldsymbol{q}||\end{bmatrix}^\top$;
4 $U_K, \Sigma_K, V_K \leftarrow$ the full SVD of K;
5 $U_A^1, \Sigma_A^1, V_A^1 \leftarrow$ the first k columns of $[U_A^0\ P]U_K, \Sigma_K, [V_A^0\ Q]V_K$.

Algorithm 2: Incremental update of $U^{(\tau)}$, $\Sigma^{(\tau)}$, and $V^{(\tau)}$

Input: $\tilde{R}^{(\tau)}, \boldsymbol{d}_U^{(\tau)}, \boldsymbol{d}_I^{(\tau)}, U^{(\tau)}, \Sigma^{(\tau)}, V^{(\tau)}, (u, i, t^{(\tau+1)})$
Output: $\tilde{R}^{(\tau+1)}, \boldsymbol{d}_U^{(\tau+1)}, \boldsymbol{d}_I^{(\tau+1)}, U^{(\tau+1)}, \Sigma^{(\tau+1)}\ V^{(\tau+1)}$

Update $\tilde{R}^{(\tau)}$ and get $\tilde{R}^{(\tau+1)}$ though Eq (1);
Update $\boldsymbol{d}_U^{(\tau)}, \boldsymbol{d}_I^{(\tau)}$ and get $\boldsymbol{d}_U^{(\tau+1)}, \boldsymbol{d}_I^{(\tau+1)}$ though Eq (3);
$\boldsymbol{\Delta}_u \leftarrow (\boldsymbol{d}_U^{(\tau+1)}[u] - \boldsymbol{d}_U^{(\tau)}[u]) \cdot \tilde{R}^{(\tau)}[u, \cdot] \odot \boldsymbol{d}_I^{(\tau)}$;
$\boldsymbol{\Delta}_i \leftarrow (\boldsymbol{d}_I^{(\tau+1)}[i] - \boldsymbol{d}_I^{(\tau)}[i]) \cdot \tilde{R}^{(\tau)}[\cdot, i] \odot \boldsymbol{d}_U^{(\tau)}$;
$\gamma_u \leftarrow (\boldsymbol{d}_U^{(\tau+1)}[u] - \boldsymbol{d}_U^{(\tau)}[u]) \cdot \tilde{R}^{(\tau)}[u, i] \cdot \boldsymbol{d}_I^{(\tau)}[i]$;
$\gamma_i \leftarrow (\boldsymbol{d}_I^{(\tau+1)}[i] - \boldsymbol{d}_I^{(\tau)}[i]) \cdot \tilde{R}^{(\tau)}[u, i] \cdot \boldsymbol{d}_U^{(\tau)}[u]$;
$\gamma_{ui} \leftarrow \boldsymbol{d}_U^{(\tau+1)}[u] \cdot (\tilde{R}^{(\tau)}[u, i] + \sigma_t(t^{(\tau+1)})) \cdot \boldsymbol{d}_I^{(\tau+1)}[i] - \boldsymbol{d}_U^{(\tau)}[u] \cdot \tilde{R}^{(\tau)}[u, i] \cdot \boldsymbol{d}_I^{(\tau)}[i]$;
$\Delta_{ui} \leftarrow \gamma_{ui} - \gamma_u - \gamma_i$;
$U_0^{(\tau+1)}, \Sigma_0^{(\tau+1)}, V_0^{(\tau+1)} \leftarrow iSVD(U^{(\tau)}, \Sigma^{(\tau)}, V^{(\tau)}, \boldsymbol{e}_u^{|\mathcal{U}|}, \boldsymbol{\Delta}_u)$;
$U_1^{(\tau+1)}, \Sigma_1^{(\tau+1)}, V_1^{(\tau+1)} \leftarrow iSVD(U_0^{(\tau+1)}, \Sigma_0^{(\tau+1)}, V_0^{(\tau+1)}, \boldsymbol{\Delta}_i, \boldsymbol{e}_i^{|\mathcal{I}|})$;
$U^{(\tau+1)}, \Sigma^{(\tau+1)}, V^{(\tau+1)} \leftarrow iSVD(U_1^{(\tau+1)}, \Sigma_1^{(\tau+1)}, V_1^{(\tau+1)}, \boldsymbol{e}_u^{|\mathcal{U}|}, \Delta_{ui} \cdot \boldsymbol{e}_i^{|\mathcal{I}|})$;

3.2.2 The Update of Sequential Score. The incremental update of $\boldsymbol{l}_u^{(\tau+1)}$ and $Q^{(\tau+1)}$ has been introduced in Section 3.2.1, user short-term interests vector $\boldsymbol{s}_u^{(\tau+1)}$ can be obtained by following Eq. (8). Then, following Eq. (9), the updated sequential score at $t^{(\tau+1)}$ is:

$$\boldsymbol{ps}_u^{(\tau+1)} = \boldsymbol{s}_u^{(\tau+1)} (Q^{(\tau+1)})^\top. \tag{15}$$

3.2.3 The Update of Transition Score. The occurrence of new interaction $(u, i, t^{(\tau+1)})$ means that there is a new transition $(i_0, t_0) \xrightarrow{\mathcal{S}_u^{(\tau+1)}} (i, t^{(\tau+1)})$. The update of the transition matrix is given as follows:

$$T^{(\tau+1)} = T^{(\tau)} + \sigma_i(t_0, t^{(\tau+1)}) \cdot \boldsymbol{e}_{i_0}^{|\mathcal{I}|} (\boldsymbol{e}_i^{|\mathcal{I}|})^\top, \tag{16}$$

In Eq. (11), $\boldsymbol{pt}_u^{(\tau+1)}$ is the i-th row of the transition matrix $T^{(\tau+1)}$. Therefore, the transition score is updated as follows:

$$\boldsymbol{pt}_u^{(\tau+1)} = T^{(\tau+1)}[i, \cdot]. \tag{17}$$

4 ANALYSIS

In this section, we first introduce the GSP-based collaborative filtering method, and then analyze the relationship between the SVD-based method and the GSP-based method. For the convenience of description, we **omit the superscript** τ when referring to the current interactions.

4.1 GSP-based Collaborative Filtering

There are three steps to realize collaborative filtering through GSP [27]: 1) construct the Laplacian matrix of item-item similarity matrix $R^\top R$ or user-user similarity matrix $RR^\top$:

$$L_I = I_{|\mathcal{I}|} - R^\top R, \quad L_U = I_{|\mathcal{U}|} - RR^\top, \tag{18}$$

where I_m is an identity matrix with size $m \times m$; 2) calculate the eigendecomposition of L_I or L_U, and construct the ideal low-pass graph filter:

$$F_I = GG^\top, \quad F_U = HH^\top, \tag{19}$$

where $G \in \mathbb{R}^{|\mathcal{I}| \times k}$ and $H \in \mathbb{R}^{|\mathcal{U}| \times k}$ are matrices composed by eigenvectors corresponding to the k smallest eigenvalues of L_I and L_U by columns, respectively; and 3) the predicted interaction matrix is obtained by graph filtering:

$$\hat{R}_I = RF_I = RGG^\top, \quad \hat{R}_U = F_U R = HH^\top R. \tag{20}$$

Eq. (20) means that the graph signal is first transformed to the Fourier space, and then only the low-frequency part is retained and is finally transformed back to the original space.

4.2 Relationship between SVD-based Method and GSP-based Method

4.2.1 GSP-based Methods can be Implemented with SVD. The classical SVD-based methods calculate the low-rank approximation $\hat{R}$ as follows:

$$\hat{R} = PQ^\top = U\Sigma V^\top, \tag{21}$$

where U, Σ and V are defined in Eq. (4). Let ρ_j be the j-th largest eigenvalue of $R^\top R$ or $RR^\top$ and ρ_j satisfy $0 \le \rho_j \le 1$ [27]. According to Eq. (18), the eigenvalues of L_I and L_U are the same, and we use ω_j to represent the j-th smallest eigenvalue of L_I or L_U. Then, there are the following conclusions: 1) $\rho_j + \omega_j = 1$ and 2) the eigenvectors of L_I and L_U corresponding to ω_j are equal with the eigenvectors of $R^\top R$ and $RR^\top$ corresponding to ρ_j, respectively, $(j = 1, 2, ..., k)$. Therefore, the following equation holds:

$$G = V, \quad H = U. \tag{22}$$

Thus, Eq (20) can be written as

$$\hat{R}_I = RGG^\top = RVV^\top, \quad \hat{R}_U = HH^\top R = UU^\top R. \tag{23}$$

A stronger conclusion is as follows:

$$U\Sigma V^\top = UU^\top R = RVV^\top. \tag{24}$$

Therefore, we can conclude that **the essence of truncated SVD is an ideal low-pass graph filter**. Further, the incremental SVD in TriSIM4Rec is a *dynamic ideal low-pass graph filter*, which can mine co-occurrence information by smoothing the interaction signals dynamically.

4.2.2 Understanding SVD from the View of Graph Filtering. The user embedding and item embedding in Eq. (5) can be written as follows:

$$P = I_{|\mathcal{U}|} U\Sigma^{1/2}, \quad Q = I_{|\mathcal{I}|} V\Sigma^{1/2}. \tag{25}$$

Eq. (25) means that the essence of P and Q is to transform users' and items' one-hot signals to the Fourier space defined by the user-user and item-item similarity graph respectively, and then do scaling transformation to transforms users and items to the same Fourier space after omitting the high-frequency part. Fourier bases correspond to different frequencies, and each dimension of users and items in Fourier space represents their score in the corresponding frequency component, so **user embedding and item embedding obtained by SVD has global structure information**. This explains why SVD is effective in collaborative filtering tasks.

5 EXPLAINABILITY AND INTERACTIVITY

TriSIM4Rec can dynamically capture user interests, which can better prevent users from being in the information cocoons [30] than static methods. However, there are still concerns that users' new interactions suffer from the exposure bias [6] of the recommendation model. To alleviate the phenomenon of information cocoons,

we extend TriSIM4Rec to an explainable and interactive method, which makes it possible for users to break the cocoons actively. We first propose the concept of *interest vector space* to introduce the explainability of TriSIM4Rec, and then analyze it from the perspective of the spectral domain to introduce its interactivity.

5.1 Interest Vector Space

A core problem in modeling users is how to represent users. Without side information, a basic assumption is that the interaction pattern between users and items can completely depict users. An *item vector space* is constructed in the spatial domain through R, in which each user is represented by an n-dimensional vector (a row of R). Each dimension of a user vector corresponds to an item, and its value is the score of the user's interaction with the item, that is, the weight after time decay (Eq. (1)) and popularity deviation (Eq. (2)). In item vector space, the similarity between two users is measured by the co-occurrence pattern of the items they interact with.

The similarity between two users can also be reflected in the similarity of their interests. A user has several interests, and each interest can be expressed by a group of similar items. We can construct the *interest vector space* through linear transformation, and map users' representation in the item vector space into the interest vector space, where the linear transformation is embodied in the form of SVD. As mentioned above, **SVD maps users and items to the same Fourier space, so P and Q have global structure information, while items that can reflect similar interests have a closer link structure on the graph, so SVD can mine the interest-based relationship between users and items**.

In TriSIM4Rec, we use P to represent users, each row of which represents a user by a k-dimensional vector in the interest vector space. Each dimension of the user vector corresponds to an interest, and its value is the user's preference for the interest. So, P can be understood as the *user-interest matrix*, which describes the distribution of users' interests. Similarly, Q can be understood as the *item-interest* matrix. It means that there are a total of k interests, and each interest is defined as a n-dimensional vector, where each dimension corresponds to one item. A larger value of a dimension in the interest vector means higher importance of the corresponding item in the interest.

Eq. (21) indicates that R can be approximated by the product of the user-interest matrix P and the item-interest matrix $Q^\top$, which means that the reconstructed user vectors in item vector space are a linear combination of k interest vectors, and the coefficient is users' representation in interest vector space. That is, a user's representation in the item vector space can be approximated by the user's vector in the interest vector space.

5.2 Frequency Analysis

The connection between SVD-based and GSP-based collaborative filtering methods enables the understanding of user interests in the spectral domain. The low-rank approximation $\hat{R}$ in Eq. (21) can be written as a weighted sum of several rank-1 matrices:

$$\hat{R} = \sum_{j=1}^{k} s_j \boldsymbol{u}_j \boldsymbol{v}_j^\top. \tag{26}$$

From the spectral domain, the essence of each rank-1 matrix is the similarity between users and items at the corresponding frequency.

Table 1: Statistics of the datasets. "Multiple" means that a user can interact with an item more than one time.

Datasets	# Users	# Items	# Interactions	Multiple
Video	5,130	1,685	37,126	✗
Game	24,303	10,672	231,780	✗
ML-100K	943	1,349	99,287	✗
ML-1M	6,040	3,416	999,611	✗
Wikipedia	8,227	1,000	157,474	✓
LastFM	980	1,000	1,293,103	✓

Therefore, the interest vector space is a concept in the spectral domain, which means that a frequency in the spectral domain corresponds to an interest in the interest vector space. So, $\hat{R}$ is the sum of different interests, and Eq. (26) disassembles the k interests.

If each user is able to change the weights of frequencies, where each frequency corresponds to a certain interest and each interest corresponds to a group of items, the users can actively control the recommendation results to break the information cocoons through an interactive recommendation.

5.3 Information Cocoons vs. Explainable and Interactive Recommendation

Explainability is the premise of interactivity, otherwise, the users cannot know the consequences of their efforts. The interest vector space allows us to explain the user representation in Fourier space with user interest, so we first explore "what is the relationship between signal frequency and user interest". After that, users can actively control the proportion of various interests to regulate the recommendation results toward their desired directions. Furthermore, user embedding is a user's representation in the interest vector space, which can express the user's preferences in different interest domains. So, we can find "how do users' long-term interests and short-term interests change dynamically" by analyzing $\boldsymbol{l}_u^{(\tau)}$ and $\boldsymbol{s}_u^{(\tau)}$ over time. We will explore these two RQs and "why is this item recommended", which is helpful to improve user satisfaction and refine the recommendation algorithm, in the experiment section.

6 EXPERIMENTS

6.1 Settings

6.1.1 Datasets. We use the Amazon Video (Video), Amazon Game (Game) [11], MovieLens-1M (ML-1M), and MovieLens-100K (ML-100K) [10] for the future item recommendation task, in which a user interacts with an item only once at most. We use Wikipedia and LastFM [18] for the next interaction prediction task, in which a user may interact with an item multiple times. We split the data by time. The first 80% interactions are for training, the following 10% interactions are for validation, and the last 10% interactions are for testing. The statistics of the datasets are shown in Table 1.

6.1.2 Metrics. We use *MRR* and *HR@K* to evaluate the performance of models on the two recommendation tasks:

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{r_i}, \quad HR@K = \frac{1}{N}\sum_{i=1}^{N} f_K(r_i). \tag{27}$$

Table 2: Comparison on future item recommendation task.

	Video	Game	ML-100K	ML-1M
	HR@10	HR@10	HR@10	HR@10
LightGCN	0.036	0.026	0.025	0.029
Time-LSTM	0.044	0.020	0.058	0.033
RRN	0.068	0.029	0.065	0.043
DeepCoevolve	0.050	0.027	0.069	0.030
JODIE	0.078	0.035	0.074	0.035
CoPE	0.088	0.047	0.081	0.049
FreeGEM	0.113	0.050	0.114	0.053
TriSIM4Rec	**0.149**	**0.052**	**0.200**	**0.161**
Relative imp. (%)	31.9%	4.0%	75.4%	203.8%

N is the number of interactions, r_i refers to the predicted ranking position of the ground truth item in the i-th interaction, and $f_K(r_i) = 1$ if $r_i \leq K$ and 0 otherwise. In this paper, we take $K = 10$.

6.1.3 Compared Methods. For the future item recommendation task, we compare TriSIM4Rec with the following methods: (1) LightGCN [12], which is a classic collaborative filtering method based on GNN, ignoring the time information. (2) RRN [37], which models user and item interaction sequences with separate RNNs. (3) Time-LSTM [43], which proposes time gates to represent the time intervals. (4) DeepCoevolve [7], which generates node embeddings using two intertwined RNNs. (5) JODIE [18], which can further estimate user embedding trajectories, compared with DeepCoevolve. (6) CoPE [41], which uses an ordinary differential equation-based GNN to model the evolution of the network. (7) FreeGEM [21], which devises an Online-Monitor-Offline architecture to model users and items dynamically. For the next interaction prediction task, we compare TriSIM4Rec with the other four methods: (1) LatentCross [1], which is a sequential recommendation method that incorporates contextual data into embeddings. (2) CTDNE [23], which is a temporal network embedding method. (3) HILI [5], which makes interaction information highly liquid to avoid information asymmetry. (4) Last-10, which is an intuitive baseline that takes the recent 10 items that a user interacted with as predictions.

6.1.4 Hyper-parameters Settings. The dimension k is searched in the range of {64, 128, 256, 512}. The time decay coefficient β_t and interval decay coefficient β_i are searched in the range of {0, 10, 20, 30, 40, 50}. The short-term interests coefficient λ_s and item transition information coefficient λ_t are searched in the range of {0.0, 0.1, ..., 1.0}. For all experiments, we report the results on the test set when the models achieve the optimal results on the verification set.

6.2 Performance Comparison

6.2.1 Future Item Recommendation. We use this task to verify whether TriSIM4Rec can accurately predict a user's future interactions based on the user's history interactions, which is a typical application of dynamic interaction graphs in recommender systems. In this task, a user interacts with an item only once at most.

As shown in Table 2, TriSIM4Rec achieves better accuracy than all compared methods on all datasets. Compared to all compared methods, the main advantage of TriSIM4Rec is that it utilizes three kinds of structural information simultaneously. Among all compared methods, LightGCN performs the worst, because it is the only static GNN model which cannot capture the dynamic characteristics of the interaction graph.

Table 3: Comparison on next link prediction task.

	Wikipedia		LastFM	
	MRR	HR@10	MRR	HR@10
Last-10	0.792	0.842	0.139	0.263
Time-LSTM	0.247	0.342	0.068	0.137
RRN	0.522	0.617	0.089	0.182
LatentCross	0.424	0.481	0.148	0.227
CTDNE	0.035	0.056	0.010	0.010
DeepCoevolve	0.515	0.563	0.019	0.039
JODIE	0.746	0.822	0.195	0.307
CoPE	0.750	**0.890**	0.200	0.446
HILI	0.761	0.853	0.252	0.427
FreeGEM	0.786	0.852	0.195	0.453
TriSIM4Rec	**0.813**	0.881	**0.346**	**0.512**
Relative imp. (%)	2.7%	-1.0%	37.3%	13.0%

Table 4: Results of the ablation study. Note that A1 to A4 have covered all valid variants of TriSIM4Rec due to the dependency between "C" and "S".

	C	T	S	Wikipedia		LastFM	
				HR@10	MRR	HR@10	MRR
A1	✗	✓	✗	0.858	0.804	0.451	0.321
A2	✓	✗	✗	0.771	0.555	0.341	0.151
A3	✓	✓	✗	0.875	0.809	0.486	0.336
A4	✓	✗	✓	0.869	0.708	0.374	0.177
TriSIM4Rec	✓	✓	✓	0.881	0.813	0.512	0.346

6.2.2 Next Interaction Prediction. We use this task to verify whether TriSIM4Rec can accurately predict users' next interaction according to the historical interactions of users, which is a kind of user behavior prediction problem. In this task, a user may interact with an item multiple times.

The results are presented in Table 3. Since JODIE, CoPE, HILI, FreeGEM and TriSIM4Rec can update models in test time, they significantly outperform the other methods without test time training. Interestingly, we found that most methods are not even better than the baseline method — Last-10. The excellent results of Last-10 on Wikipedia are due to the repeated interactions between users and items in Wikipedia. As users interact with the same item on LastFM at a relatively lower frequency, Last-10 performs poorly, but some methods still fail to outperform the performance of Last-10. In addition, on LastFM, we noticed that the HRs of FreeGEM and CoPE were improved by about 50% compared with JODIE, but their MRRs were hardly improved. Fortunately, TriSIM4Rec has greatly improved in MRR. TriSIM4Rec increases the HR by 13.0% compared with FreeGEM, while the MRR increases by 37.3% compared with HILI. We will further explore the mechanism that affects HR and MRR later in Section 6.5.1.

6.3 Ablation Study

As shown in Table 4, we use A1, A2, A3, A4 to refer to ablative variants of TriSIM4Rec, "C", "T", and "S" to refer to co-occurrence

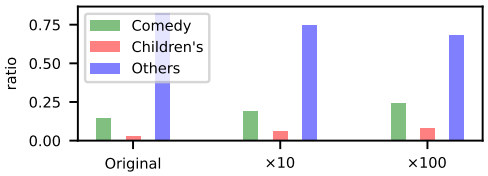

Figure 2: The impact of amplifying the frequency corresponding to the comedy.

information, item transition information, and sequential information respectively, and check or cross marks indicate whether the corresponding information exists. We can see that when one of the co-occurrence information, item transition information and sequential information is not adopted, the results are suboptimal, which confirms the effectiveness of all three structural information.

Since the attention module of modeling sequential information in TriSIM4Rec is based on user embedding and item embedding obtained through modeling co-occurrence information, sequential information cannot exist without co-occurrence information. Therefore, there is no valid variant of TriSIM4Rec besides A1 to A4.

6.4 Explainability and Interactivity

We discuss three RQs through case studies and statistics to validate the explainability and interactivity of TriSIM4Rec. The experiments are conducted on the ML-1M dataset.

6.4.1 RQ1: Why is This Item Recommended? In Eq. (20), from the perspective of graph filtering, the items that a user has interacted with have strong signals (>0), while the items that the user has not interacted with have weak signals (=0). As the recommended items have not been interacted with by the user, they have low-intensity signals before filtering, but after filtering, they are pulled up by the items the user has interacted with. This means that the score of each recommended item is only affected by the items that the user has interacted with. Specifically, the score of each recommended item is the sum of the similarity between the item and other items that the user has interacted with in the interest vector space.

We take the most three similar items that a user has interacted with as explanations based on the similarity of items in interest vector space. Table 5 shows three recommendation results of user 1081 in three time periods, and three explanations corresponding to each recommendation result. It can be inferred from the recommendation results that this user prefers comedy movies in the early stage, and sci-fi and war movies in the later stage. And each movie as an explanation has been watched by this user before.

6.4.2 RQ2: What Is the Relationship between Signal Frequency and User Interest? As shown in Table 6, we list the representative movies of different interest domains, that is, three items with the highest scores on the corresponding frequency. It can be seen that the leading movie genre of the 3rd frequency is comedy, and the leading movie genre of the 4th frequency is sci-fi and war. We amplify the intensity of the 3rd frequency by 10 times and 100 times larger than the original. Figure 2 shows the proportion of comedy, children's and other types of movies in the original recommendation results,

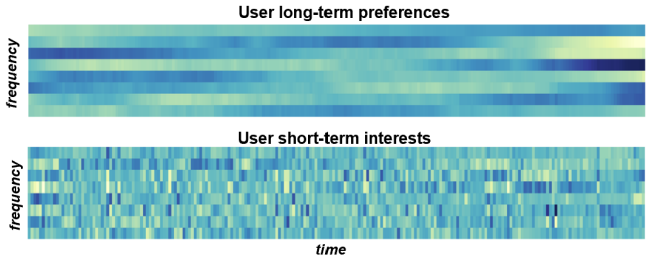

Figure 3: The change of long-term interests and short-term interests of user 1081 over time.

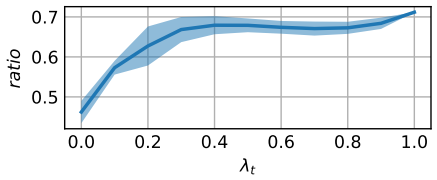

Figure 4: Impact of item transition information and co-occurrence information on recommendation results.

and the proportion of three types of movies after two amplifications. It can be seen that, after amplification, the proportion of comedy movies increased in the recommendation results, and the children's movies also increased due to a high correlation with comedy. This shows that users can control their own recommendation results by controlling the weights of different frequencies.

The results of RQ2 show that users can clearly know what kind of interest a certain frequency represents. When users are dissatisfied with the default recommendation results, they can actively interact with the recommendation model to regulate the recommendation results toward their desired directions.

6.4.3 RQ3: How Do Users' Long-Term Interests and Short-Term Interests Change Dynamically? Figure 3 shows how user 1081's long-term interests vector and short-term interests vector change over time. Darker color means a higher score. It can be seen that the user's long-term interests change smoothly over time, while the user's short-term interests change dramatically over time. Through the user's long-term interests, we observe that: 1) the user has a higher score on the 3rd frequency in the early stage, indicating that she/he prefers comedy at this stage; and 2) the user has a higher score on the 4th frequency in the later stage, indicating that she/he prefers sci-fi and war movies at this stage. The observation is consistent with the analyses in RQ1 and RQ2.

6.5 Sensitivity, Efficiency and Robustness

6.5.1 Impacts of Co-occurrence and Item transition Information. We conduct this experiment on LastFM. Firstly, we calculate the $ratio$ for the results of each group of hyper-parameters:

$$ratio = MRR/HR. \tag{28}$$

Then we group the results by λ_t and calculate the mean value and the standard deviation of $ratio$ of each group. As shown in Figure

Table 5: Recommendation results and explanations of user 1081 for different time periods.

	Period 1	Period 2	Period 3
Recommended 1	**Modern Times**	**The Shawshank Redemption**	**Edward Scissorhands**
Explain 1-1	You Can't Take It With You	Evita	Who Framed Roger Rabbit?
Explain 1-1	Mary Poppins	Inherit the Wind	The Deep End of the Ocean
Explain 1-3	Manhattan	The Great Race	The King and I
Recommended 2	**Back to the Future Part II**	**Modern Times**	**The X-Files: Fight the Future**
Explain 2-1	A Simple Plan	You Can't Take It With You	Brokedown Palace
Explain 2-2	Philadelphia	Mary Poppins	The Lost Weekend
Explain 3-3	Babe	Manhattan	Heavenly Creatures
Recommended 3	**Gattaca**	**The X-Files: Fight the Future**	**Saving Private Ryan**
Explain 3-1	The Lost Weekend	Brokedown Palace	Entrapment
Explain 3-2	Heavenly Creatures	The Lost Weekend	Inherit the Wind
Explain 3-3	The Crying Game	Heavenly Creatures	The Big Sleep

Table 6: Leading movies with different frequencies (domains of interest).

Frequency	The 1st leading movie	The 2nd leading movie	The 3rd leading movie
1	Laura	Murder, My Sweet	Duel in the Sun
2	The General's Daughter	Double Jeopardy	Austin Powers: The Spy Who Shagged Me
3	Austin Powers: The Spy Who Shagged Me	A Bug's Life	Doctor Dolittle
4	Star Wars V	Star Wars IV	Star Wars VI
5	While You Were Sleeping	Sleepless in Seattle	My Best Friend's Wedding
6	Erin Brockovich	American Beauty	American Pie
7	Annie Hall	Chinatown	Casablanca
8	Midaq Alley (Callejón de los Milagros, El)	In God's Hands	Return with Honor

Table 7: Running time comparison.

	Wikipedia	LastFM
JODIE	7m13s (per epoch)	221m48s (per epoch)
CoPE	106m52s (per epoch)	1,471m2s (per epoch)
FreeGEM	11m31s	51m56s
TriSIM4Rec	19m31s	99m15s

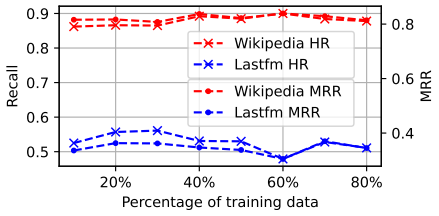

Figure 5: Robustness analysis of TriSIM4Rec.

4, the blue line is the mean value, and the light blue shadow is the triple standard deviation.

With the increase of λ_t, item transition information will increase and co-occurrence information will decrease, we find that the *ratio* will increase. This means that item transition information is more conducive to the rise of MRR, a ranking-related metric, while co-occurrence information is more conducive to the rise of HR, a recall-related metric.

6.5.2 Running time. We use the next interaction prediction task to study the efficiency of JODIE, CoPE, FreeGEM and TriSIM4Rec. JODIE and CoPE both need to run multiple epochs and choose the best-performing model on the validation set as the optimal model. We measure the time it takes to run a single epoch and their total running time should multiply by a constant. As shown in Table 7, The efficiency of TriSIM4Rec is much higher than that of JODIE and CoPE. Compared with FreeGEM, the efficiency of TriSIM4Rec is worse due to 1) the introduction of item transition information and 2) we update user embedding and item embedding more accurately by disassembling a rank-2 matrix into three rank-1 matrices. Although these operations increase the time complexity, they also improve the prediction accuracy significantly.

6.5.3 Robustness Studies. For the next interaction prediction task, we change the proportion of the training set to verify the robustness of TriSIM4Rec in different levels of data sparsity. We change the percentage of the training set from 10% to 80%, the next 10% interactions after the training set as the validation set, and next the 10% interactions after the validation set as the test set. The results are shown in Figure 5, in which we can observe that the accuracy of TriSIM4Rec is almost unaffected. This experiment demonstrates that TriSIM4Rec has strong robustness to the scale of training data.

7 CONCLUSION

In this paper, we propose TriSIM4Rec to solve the recommendation tasks on the dynamic graph by using three types of structural information in user-item interaction data. Incremental SVD enables TriSIM4Rec to dynamically and incrementally model users and items. Then, we analyze the relationship between the classical SVD-based and the recently emerging GSP-based collaborative filtering algorithms. Finally, we extend TriSIM4Rec to an explainable and interactive recommendation method. Extensive experiments on various datasets demonstrate the effectiveness of TriSIM4Rec.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (NSFC) under Grants 61932007 and 62172106.

REFERENCES

[1] Alex Beutel, Paul Covington, Sagar Jain, Can Xu, Jia Li, Vince Gatto, and Ed H Chi. 2018. Latent cross: Making use of context in recurrent recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 46–54.

[2] Matthew Brand. 2006. Fast low-rank modifications of the thin singular value decomposition. *Linear algebra and its applications* 415, 1 (2006), 20–30.

[3] Jiangxia Cao, Xixun Lin, Xin Cong, Shu Guo, Hengzhu Tang, Tingwen Liu, and Bin Wang. 2021. Deep structural point process for learning temporal interaction networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 305–320.

[4] Xiaofu Chang, Xuqin Liu, Jianfeng Wen, Shuang Li, Yanming Fang, Le Song, and Yuan Qi. 2020. Continuous-time dynamic graph learning via neural interaction processes. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 145–154.

[5] Huidi Chen, Yun Xiong, Yangyong Zhu, and Philip S Yu. 2021. Highly liquid temporal interaction graph embeddings. In *Proceedings of the Web Conference 2021*. 1639–1648.

[6] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and debias in recommender system: A survey and future directions. *arXiv preprint arXiv:2010.03240* (2020).

[7] Hanjun Dai, Yichen Wang, Rakshit Trivedi, and Le Song. 2016. Deep coevolutionary network: Embedding user and item features for recommendation. *arXiv preprint arXiv:1609.03675* (2016).

[8] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM* 35, 12 (1992), 61–70.

[9] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. 2009. Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. (2009).

[10] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[11] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*. 507–517.

[12] 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.

[13] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[14] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.

[15] Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. 2020. Representation Learning for Dynamic Graphs: A Survey. *J. Mach. Learn. Res.* 21, 70 (2020), 1–73.

[16] Zekarias Kefato, Sarunas Girdzijauskas, Nasrullah Sheikh, and Alberto Montresor. 2021. Dynamic embeddings for interaction prediction. In *Proceedings of the Web Conference 2021*. 1609–1618.

[17] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[18] Srijan Kumar, Xikun Zhang, and Jure Leskovec. 2019. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 1269–1278.

[19] Dongsheng Li, Chao Chen, Wei Liu, Tun Lu, Ning Gu, and Stephen Chu. 2017. Mixture-rank matrix approximation for collaborative filtering. *Advances in Neural Information Processing Systems* 30 (2017).

[20] Xiaohan Li, Mengqi Zhang, Shu Wu, Zheng Liu, Liang Wang, and S Yu Philip. 2020. Dynamic graph collaborative filtering. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, 322–331.

[21] Jiahao Liu, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, and Ning Gu. 2022. Parameter-free Dynamic Graph Embedding for Link Prediction. *arXiv preprint arXiv:2210.08189* (2022).

[22] Jiahao Liu, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, Li Shang, and Ning Gu. 2023. Personalized Graph Signal Processing for Collaborative Filtering. *arXiv preprint arXiv:2302.02113* (2023).

[23] Giang Hoang Nguyen, John Boaz Lee, Ryan A Rossi, Nesreen K Ahmed, Eunyee Koh, and Sungchul Kim. 2018. Continuous-time dynamic network embeddings. In *Companion Proceedings of the The Web Conference 2018*. 969–976.

[24] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2000. *Application of dimensionality reduction in recommender system-a case study*. Technical Report. Minnesota Univ Minneapolis Dept of Computer Science.

[25] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[26] Oleksandr Shchur, Marin Biloš, and Stephan Günnemann. 2019. Intensity-free learning of temporal point processes. *arXiv preprint arXiv:1909.12127* (2019).

[27] 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.

[28] Harald Steck. 2011. Item popularity and recommendation accuracy. In *Proceedings of the fifth ACM conference on Recommender systems*. 125–132.

[29] Harald Steck. 2019. Markov random fields for collaborative filtering. *Advances in Neural Information Processing Systems* 32 (2019).

[30] Cass R Sunstein. 2006. *Infotopia: How many minds produce knowledge*. Oxford University Press.

[31] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 565–573.

[32] Sheng Tian, Tao Xiong, and Leilei Shi. 2021. Streaming Dynamic Graph Neural Networks for Continuous-Time Temporal Graph Modeling. In *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1361–1366.

[33] Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. 2017. Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. In *international conference on machine learning*. PMLR, 3462–3471.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[35] Yichen Wang, Nan Du, Rakshit Trivedi, and Le Song. 2016. Coevolutionary latent feature processes for continuous-time user-item interactions. *Advances in neural information processing systems* 29 (2016).

[36] Zhihao Wen and Yuan Fang. 2022. TREND: TempoRal Event and Node Dynamics for Graph Representation Learning. In *Proceedings of the ACM Web Conference 2022*. 1159–1169.

[37] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. 2017. Recurrent recommender networks. In *Proceedings of the tenth ACM international conference on web search and data mining*. 495–503.

[38] 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.

[39] Cheng Yang, Chunchen Wang, Yuanfu Lu, Xumeng Gong, Chuan Shi, Wei Wang, and Xu Zhang. 2022. Few-shot Link Prediction in Dynamic Networks. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 1245–1255.

[40] Yao Zhang, Yun Xiong, Xiangnan Kong, and Yangyong Zhu. 2017. Learning node embeddings in interaction graphs. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 397–406.

[41] Yao Zhang, Yun Xiong, Dongsheng Li, Caihua Shan, Kan Ren, and Yangyong Zhu. 2021. CoPE: Modeling Continuous Propagation and Evolution on Interaction Graph. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2627–2636.

[42] Zhen Zhang, Jiajun Bu, Martin Ester, Jianfeng Zhang, Chengwei Yao, Zhao Li, and Can Wang. 2020. Learning temporal interaction graph embedding via coupled memory networks. In *Proceedings of the web conference 2020*. 3049–3055.

[43] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to Do Next: Modeling User Behaviors by Time-LSTM.. In *IJCAI*, Vol. 17. 3602–3608.

[44] Yuan Zuo, Guannan Liu, Hao Lin, Jia Guo, Xiaoqian Hu, and Junjie Wu. 2018. Embedding temporal network via neighborhood formation. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2857–2866.