Spectrum-enhanced Pairwise Learning to Rank

Wenhui Yu
Tsinghua University
Beijing, China
yuwh16@mails.tsinghua.edu.cn

Zheng Qin*
Tsinghua University
Beijing, China
qingzh@mail.tsinghua.edu.cn

ABSTRACT

To enhance the performance of the recommender system, side information is extensively explored with various features (e.g., visual features and textual features). However, there are some demerits of side information: (1) the extra data is not always available in all recommendation tasks; (2) it is only for items, there is seldom high-level feature describing users. To address these gaps, we introduce the *spectral features* extracted from two hypergraph structures of the purchase records. Spectral features describe the *similarity* of users/items in the graph space, which is critical for recommendation. We leverage spectral features to model the users' preference and items' properties by incorporating them into a Matrix Factorization (MF) model.

In addition to modeling, we also use spectral features to optimize. Bayesian Personalized Ranking (BPR) is extensively leveraged to optimize models in *implicit feedback* data. However, in BPR, all missing values are regarded as negative samples equally while many of them are indeed unseen positive ones. We enrich the positive samples by calculating the similarity among users/items by the spectral features. The key ideas are: (1) similar users shall have similar preference on the same item; (2) a user shall have similar perception on similar items. Extensive experiments on two real-world datasets demonstrate the usefulness of the spectral features and the effectiveness of our spectrum-enhanced pairwise optimization. Our models outperform several state-of-the-art models significantly.

CCS CONCEPTS

• **Information systems** → **Collaborative filtering**; **Social recommendation**; **Recommender systems**; • **Human-centered computing** → *Social recommendation.*

KEYWORDS

Collaborative filtering, spectral feature, pairwise learning to rank, latent community, latent category.

ACM Reference Format:
Wenhui Yu and Zheng Qin. 2019. Spectrum-enhanced Pairwise Learning to Rank. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313478

*The corresponding author.

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313478

1 INTRODUCTION

Recommender systems have been widely used in online services such as E-commerce and social media sites to predict users' preference based on their interaction histories. Modern recommender systems uncover the underlying latent factors that encode the preference of users and properties of items. In recent years, to strengthen the presentation ability of the models, various features are incorporated for additional information, such as visual features from product images [14, 44, 48], textual features from review data [8, 9], and auditory features from music [5]. However, these features are not generally applicable, for example, visual features can only be used in product recommendation, while not in music recommendation. Also, these features are unavailable in some recommendation tasks, such as point-of-interest recommendation [43, 45]. Even they are available, we need extra efforts to collect and process them. Moreover, we can only get features for items while there is seldom high-level feature for users. To address these gaps, we introduce spectral features extracted from the purchase records for both items and users in this paper. We then use the spectral features to (1) model users' preference and items' properties and to (2) optimize the proposed model.

Recommender systems predict the missing value by uncovering the similarity of users/items, thus the information of similarity is vital in recommendation tasks. For example, in user-/item-based collaborative filtering, we recommend by calculating the similarity among users/items [12, 39]; in model-based collaborative filtering, the low-rank form ensures the linear dependence of latent factors, i.e., the similarity among users/items [1, 22, 35]. Inspired by this, we propose a new feature that contains the similarity information. We define the front K eigenvectors of a Laplacian matrix as the spectral feature. Devised for spectral clustering [29], the spectral feature describes the distance among vertices in a graph space thus contains abundant information of similarity. The spectral feature extracts information from the purchase history rather than additional data, thus it is generally applicable and works in almost all situations. Also, we can extract spectral features for both users and items.

We define the set of users near in the graph space as a *latent community*, which means they have similar purchase behaviors. For an item preferred by a specific user, it may get a high score for her latent community members. Also, the set of items near in the graph space is defined as a *latent category*, items in the same latent category have similar properties. Users who like certain item may have interests in the latent category the item belongs to. Of special notice is that a latent category is not like a real category, items in it may be relevant items, similar items, etc., in a word, items strongly connected in the graph. In this paper, we incorporate the spectral features into a Matrix Factorization (MF) model [22, 37] to propose our **S**pectrum-enhanced **C**ollaborative **F**iltering (**SCF**) model. By

learning spectral dimensions, SCF uncovers users' preference for latent categories and items' fitness for latent communities.

Compared with other side information features, our spectral feature is more suitable for the *implicit feedback* data. Implicit feedback data is like "purchase" or "browse" in E-commerce sites, "like" in social media sites, "click" in advertisements, etc. In real-world applications, the data of user behaviours in "one-class" (implicit feedback) is easier to collect and more generally applicable than "multi-class" scores (*explicit feedback*). The spectral features, which are independent of extra data, are more suitable for the implicit feedback data, since they maintain the advantages of easy to collect and generality.

Besides providing information of similarity, we also use the spectral features to enhance the pairwise learning. When optimizing model on implicit feedback data, Bayesian Personalized Ranking (BPR) is widely used due to the outstanding performance [14, 37, 44]. It aims to maximize the likelihood of pairwise preference over positive samples and negative samples. However, there is a critical issue: All missing values are simply treated as negative samples in BPR. In fact, some of the missing values are indeed unlabelled positive samples, users may like them but just have not seen them yet. To deal with this issue, we cluster all items and users by spectral features to construct the latent categories and latent communities, respectively. We assume that a user shows stronger preference for items that are in the same latent category with what she purchased, or items that purchased by her neighbors in the same latent community, than other missing entries. Considering the enriched preference relationship, we propose an optimization method called **S**pectrum-enhanced **P**airwise **L**earning to **R**ank (**SPLR**), and optimize SCF with it. Finally, we validate the effectiveness of our proposed model by comparing it with several baselines on the *Amazon.Clothes* and *Amazon.Jewelry* datasets. Extensive experiments show that we improve the performance significantly by exploring spectral features.

Specifically, our main contributions are listed as follows:

- We leverage novel spectral features in recommendation tasks to capture the similarity information of users/items, and propose an SCF model by injecting these features to the MF model.
- We propose a spectral clustering-enhanced pairwise ranking method, SPLR, to optimize our model. We construct latent categories and latent communities to enrich the positive samples.
- We extend our model and propose a framework to use side information features for modeling and for optimization. We can explore any kinds of features in our framework, including spectral features and other conventional features. Multiple features can be utilized at a time.
- We devise comprehensive experiments on two real-world datasets to demonstrate the effectiveness of our proposed methods.

2 RELATED WORK

After Matrix Factorization (MF) is utilized to deal with recommendation tasks [1, 22, 37], modern recommender systems develop rapidly. MF models learn the latent factors of users and items by reconstructing the purchase records in a low-rank form. Latent factors represent the preference of users and properties of items. Many variants are proposed to promote the ability to model the complex preference that users exhibit toward items based on their past interactions. [15, 18] used deep structure to learn embedding. [7, 17] took time into account when making predictions. [19, 46] focused on fast algorithms for online recommendation.

2.1 Side Information Features

One important way to enhance the presentation ability is to leverage the side information. Theoretically, latent dimensions can capture all relevant factors, but they usually cannot in applications due to the sparsity of the datasets, thus extra information is desired. The visual features are widely used since users' decisions depend largely on products' appearance [14, 27, 44, 48]. [14, 27] predicted consumers' behavior with the CNN feature. Yu et al. [44] utilized the aesthetic feature to model users' aesthetic preference on clothes. Zhao et al. [48] leveraged several visual features to recommend movies. There are also many efforts exploring the textural feature to recommend [9, 26], McAuley and Leskovec [26] proposed models with the textural feature from the review data and Chen et al. [9] extracted the textural feature from time-synchronized comments for key-frame recommendation. In music recommendation, the auditory feature is generally used, Cao et al. [5] extracted the auditory feature by a pre-trained deep structure to recommend music.

Various features are used for different kinds of information. However, one feature, the spectral feature, has never been explored. In this paper, we extract the spectral features from the hypergraphs of users and items, which represent the similarity of vertices in the hypergraphs. Compared with features mentioned above, the spectral features are independent of additional information so they are suitable for more situations. Moreover, conventional features are all for items, while we extract spectral features for both users and items.

2.2 Graph-enhanced Recommendation

In recent years, hypergraph gains increasing attention in the recommendation domain [4, 23, 34]. Lierde and Chow [23] proposed a user-based collaborative filtering enhanced with the hypergraph: The similarity of the users is calculated with the hypergraph embedding. [4, 34] filtered the latent factors with hypergraph regularized term to smooth them. There are also some efforts promoting the recommendation performance with social networks [21, 42]. Walter et al. [42] calculated similarities with the social networks embedding for memory-based collaborative filtering. Jamali and Ester [21] calculated the latent factors of a user with its neighbors to give the prediction. Graph convolution networks are also widely used in recommendation tasks [2, 50]. Berg et al. [2] proposed a graph convolutional auto-encoder that learns the structural information of a graph for latent factors of users and items. Zheng et al. [50] constructed random walk laplacian matrix of the user-item bipartite graph and then introduced a deep spectral convolutional network to capture the user-item connectivity information.

In this paper, we introduce a new way to explore graph structures in recommendation tasks. We extract features from the Laplacian matrix of the hypergraphs of users and items, and make prediction

with the MF term jointly. We also utilize these features to optimize our proposed method.

2.3 Learning to Rank

As implicit feedback data is easier to collect, it is extensively used in real-world application. However, prediction on implicit feedback dataset is a challenging task because there are only positive samples and unobserved samples. We cannot discriminate negative samples and unlabeled positive samples from the unobserved ones. Rendle et al. [37] treated all unobserved samples as negative ones when sampling while some of them are indeed unlabelled positive samples. Users may like them but just have not seen them yet. To address this gap, many works improved the pairwise learning to rank method. It is assumed that all users are independent in BPR, Pan and Chen [30] tried to relax this constraint and proposed a method called group preference-based Bayesian personalized ranking (GBPR), which modeled the preference of user groups. Qiu et al. [33] constructed the preference chain of item groups for each user. Liu et al. [24] utilized collaborative information mined from the interactions between users and items. [36, 47] proposed dynamic negative sampling strategies to maximize the utility of a gradient step by choosing "difficult" negative samples. [10, 31] used view information to enrich positive samples. [6, 25] proposed listwise ranking methods instead of pairwise ones. Hwang et al. [20] utilized both implicit and explicit feedback data to improve the quality of negative sampling.

In existing efforts, only low-order connections are considered when measuring the similarity among vertices [24, 30, 33]. In this paper, we use the spectral features, which contains the information of high-order connections, to enhance the pairwise learning. We cluster all items/users by spectral features to construct latent categories/communities. For vertices (users/items) in the same cluster, they are strongly connected[1] in the hypergraph thus are very similar to each other. For each user, we regard items in the same latent category with her positive samples and items purchased by her latent community members as the potential samples, and assume that the user prefers them than other negative samples.

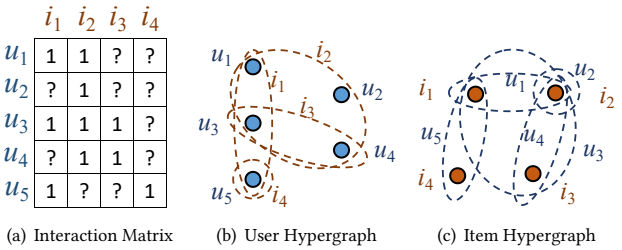
3 SPECTRUM-ENHANCED COLLABORATIVE FILTERING

In this section, we propose a novel recommendation model called Spectrum-enhanced Collaborative Filtering (SCF). We first introduce the spectral features and then inject them into an MF model. In this paper, bold uppercase letters refer to matrices. For example, $\mathbf{A}$ is a matrix, $\mathbf{A}_i$ is the i-th row of $\mathbf{A}$, $\mathbf{A}_{*,j}$ is the j-th column of $\mathbf{A}$, and $\mathbf{A}_{ij}$ is the value at the i-th row and the j-th column of $\mathbf{A}$. $\mathbf{A}^U$ is the matrix for user and $\mathbf{A}^I$ is the matrix for item, $\mathbf{A}^{(k)}$ is the k-th matrix.

3.1 Spectral Feature

In a recommendation task, we use matrix $\mathbf{R} \in \mathbb{R}^{N\times M}$ to denote the interactions between users and items (there are N users and M items in total). $\mathbf{R}_{ui} = 1$ if user u purchased items i and $\mathbf{R}_{ui} = 0$ otherwise. Our task is to predict the missing values (0 in $\mathbf{R}$) to recommend top-n items for each user.

The hypergraph is a generalization of the simple graph, where an hyperedge (edge in hypergraph) connects any number of vertices rather than just two. A hypergraph is usually represented with the incidence matrix $\mathbf{H} \in \mathbb{R}^{N\times M}$ (there are N vertices and M hyperedges). Each row in $\mathbf{H}$ is for a vertex and each column is for a hyperedge. $\mathbf{H}_{ij} = 1$ if vertex i is connected by hyperedge j and $\mathbf{H}_{ij} = 0$ otherwise. We can see that a hypergraph can represent the purchase records by nature. We define two hypergraph structures, a user hypergraph and an item hypergraph. In the user hypergraph, users are vertices and items are hyperedges while in the item hypergraph, items are vertices and users are hyperedges.



(a) Interaction Matrix (b) User Hypergraph (c) Item Hypergraph

Figure 1: An example for user hypergraph and item hypergraph. For the user hypergraph, the incidence matrix $\mathbf{H}^U = \mathbf{R}$ and for the item hypergraph, $\mathbf{H}^I = \mathbf{R}^\top$.

For a hypergraph represented with the incidence matrix $\mathbf{H}$, the Laplacian matrix $\mathbf{L}$ is defined as [51]:

$$\mathbf{L} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{D} - \mathbf{H}\mathbf{W}\mathbf{\Delta}^{-1}\mathbf{H}^\top)\mathbf{D}^{-\frac{1}{2}}, \tag{1}$$

where the diagonal matrix $\mathbf{D} \in \mathbb{R}^{N\times N}$ denotes the degrees of vertices, the diagonal matrix $\mathbf{W} \in \mathbb{R}^{M\times M}$ denotes the weights of hyperedges, and the diagonal matrix $\mathbf{\Delta} \in \mathbb{R}^{M\times M}$ denotes the degrees of hyperedges. Laplacian matrix is a difference operator of a hypergraph [28], for any signal $\mathbf{S} \in \mathbb{R}^{N\times L}$, it satisfies:

$$(\mathbf{LS})_i = \sum_{j\in\mathcal{N}_i}\sum_{k=1}^{M}\mathbf{H}_{ik}\frac{\mathbf{W}_{kk}}{\mathbf{\Delta}_{kk}}\mathbf{H}_{jk}\left(\frac{\mathbf{S}_i}{\sqrt{\mathbf{D}_{ii}}} - \frac{\mathbf{S}_j}{\sqrt{\mathbf{D}_{jj}}}\right),$$

where $\mathcal{N}_i$ is the set of vertices connected to vertex i. We can see the effect of the Laplacian matrix is to take the first-order difference in the neighborhood of vertex i. In many machine learning tasks, the parameter $\mathbf{S}$ is smoothed by minimizing the Laplacian regularization term $trace(\mathbf{S}^\top\mathbf{L}\mathbf{S})$ [11, 34].

To extract the spectral feature, we factorize the Laplacian matrix with eigen-decomposition: $\mathbf{L} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^\top$, where $\mathbf{\Lambda} = diag(\lambda_1, \lambda_2, \cdots, \lambda_N)$ is the eigenvalue matrix, all eigenvalues are in ascending order, i.e., $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_N$. We define the matrix formed by the first K eigenvectors as the spectral feature matrix: $\mathbf{F} = \mathbf{\Phi}_{*,1:K}$, and $\mathbf{F}_i$ is the spectral feature of vertex i. The spectral feature can be used for spectral clustering [23, 29] and graph Fourier transform [28]. It contains the information of the similarity among vertices thus we can measure the similarity of two vertices i and j with $\mathbf{F}_i\mathbf{F}_j^\top$. Though eigen-decomposition is computationally expensive (taking the user hypergraph as an example, with $O(N^3)$ time complexity, where N

[1] We use "strongly connected" to indicate that vertices are connected by many paths in the graph (including direct connections or high-order connections), which is different from the conception in directed graphs [40].

is the user number), however, $\mathbf{L}$ is highly sparse in recommendation tasks. Considering there are $O(N)$ non-zero elements in $\mathbf{L}$, the time complexity could be $O(K^2N)$ with Lanczos method [13, 32], which is much smaller than $O(N^3)$.

3.2 Hybrid Model

For the hybrid model (SCF), we incorporate the spectral features into a basic MF model, which is the state-of-the-art for rating prediction as well as modeling implicit feedback, to reconstruct the interaction matrix with the low-rank form:

$$\hat{\mathbf{R}} = \mathbf{U}\mathbf{V}^\mathsf{T} + \mathbf{P}\mathbf{F}^\mathsf{T} + \mathbf{E}\mathbf{Q}^\mathsf{T}, \tag{2}$$

where $\hat{\mathbf{R}} \in \mathbb{R}^{N\times M}$ is the reconstruction, $\mathbf{U} \in \mathbb{R}^{N\times K_0}$ and $\mathbf{V} \in \mathbb{R}^{M\times K_0}$ are the latent factors of users and items respectively. $\mathbf{U}_i$ is the preference of user i and $\mathbf{V}_j$ is the properties of item j. $\mathbf{E} \in \mathbb{R}^{N\times K_1}$ and $\mathbf{F} \in \mathbb{R}^{M\times K_2}$ are the spectral features of users and items respectively. $\mathbf{P} \in \mathbb{R}^{N\times K_2}$ is the preference matrix of users, $\mathbf{Q} \in \mathbb{R}^{M\times K_1}$ is the fitness matrix of items.

Considering that strongly connected users/items shall have similar preference/properties, $\mathbf{E}/\mathbf{F}$ contains the information of similarity among users/items. We use the similarity information to enhance the MF predictor. We call the three terms in Equation (2) as model-based, item-based, and user-based collaborative terms respectively. In a conventional item-based collaborative filtering model, we calculate the similarity between each pair of items. For a positive sample i and a missing sample j, if j is similar with i, we recommend j to current user. Similarly, in the second term of our predictor, for current user u and her purchased item i, $\mathbf{P}_u\mathbf{F}_i^\mathsf{T}$ is high. For a similar item j, $\mathbf{F}_i\mathbf{F}_j^\mathsf{T}$ is high, thus $\mathbf{P}_u\mathbf{F}_j^\mathsf{T}$ is high as well, and j can be recommended to u. We can see that this procedure is just the same as item-based collaborative filtering, so we call the second term of Equation (2) item-based collaborative term. And the third term, for the same reason, is called user-based collaborative term.

Now we briefly discuss the advantage of our item-based collaborative term over conventional item-based collaborative filtering. In item-based collaborative filtering, when calculating the similarity between a pair of items, only first-order connections are taken into account, while in our item-based collaborative term, high-order connections are also considered. Take Figure 1(c) as an example, i_3 and i_4 are not connected by certain hyperedge directly, hence the similarity is 0 in conventional item-based collaborative filtering. While in our model, connections i_4-i_1-i_3 and i_4-i_1-i_2-i_3 are also considered, therefore $sim(i_3, i_4) > 0$ (we will discuss the reason detailedly in the next section). In fact, in Equation (2), three models give the final prediction jointly: An MF model, and two hypergraph spectrum-enhanced memory-based collaborative filtering models.

4 SPECTRUM-ENHANCED PAIRWISE LEARNING

Besides providing information to model users' preference and items' property, spectral features are also used to optimize the model. Optimization on implicit feedback data usually takes the form of pairwise learning, which maximizes the likelihood of relative preference over a pair of positive and negative feedbacks:

$$\text{BPR\_OPT} = \sum_{u\in\mathcal{U}}\sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{I}\setminus\mathcal{I}_u^+} \ln\sigma\left(\hat{\mathbf{R}}_{uij}\right) - \frac{\lambda_r}{2}\|\mathbf{\Theta}\|_\mathrm{F}^2 .$$

where $\mathcal{U}$ and $\mathcal{I}$ are the sets of users and items. $\mathcal{I}_u^+$ is the set of positive items of u. $\sigma(\ )$ is the sigmoid function. $\hat{\mathbf{R}}$ is defined in the Equation (2) and $\hat{\mathbf{R}}_{uij} = \hat{\mathbf{R}}_{ui} - \hat{\mathbf{R}}_{uj}$. The last term is the regularization term to prevent overfitting, where λ_r is the regularization coefficient, $\|\ \|_\mathrm{F}$ is the Frobenius norm of the matrix, and $\mathbf{\Theta}$ represents the parameters of the model.

There is a critical issue: A user did not purchase an item may not because she has no interest in it, but just because she has never seen it yet. Our task is to uncover users' preference and recommend them unseen items they are interested in. However, in BPR, all missing entries are treated as negative samples nevertheless some of them are indeed unlabeled positive samples. To address this gap, we enrich the positive samples by the spectral clustering.

4.1 Objective Function

We cluster all vertices (items/users) by the normalized spectral features. Vertices in the same cluster are strongly connected in the hypergraph, though may not be connected directly.

Latent category: We define a cluster of items as a latent category, items in it are of the similar kind, or highly relevant, since they are purchased by the same user or users with similar preference. For certain item i, the latent category it belongs to is denoted as C_i. We argue that if a user likes i, she may like the items in C_i with a high probability.

Latent community: Similarly, we define a cluster of users as a latent community, users in it purchase the same items, similar items, or relevant items, thus they have similar preference. For certain user u, the latent community she belongs to is denoted as C_u. If a user likes i, her latent community members may like i as well.

To give the relative preference, we construct three sets for each user u:

$$\begin{cases} \mathcal{I}_u^+ = \{i|\mathbf{R}_{ui} = 1\}, & \text{Positive set} \\ \mathcal{P}_u = \mathcal{P}_u^U \bigcup \mathcal{P}_u^I, & \text{Potential set} \\ \mathcal{I}_u^- = \mathcal{I} - \mathcal{I}_u^+ - \mathcal{P}_u, & \text{Negative set} \end{cases}, \tag{3}$$

where $\mathcal{P}_u^U = \{i|i \in \mathcal{I}_v^+, v \in C_u \& i \notin \mathcal{I}_u^+\}$ is the user-based collaborative potential set, $\mathcal{P}_u^I = \{j|j \in C_i, i \in \mathcal{I}_u^+ \& j \notin \mathcal{I}_u^+\}$ is the item-based collaborative potential set. We have the preference relationship:

$$(u, \mathcal{I}_u^+) \succ (u, \mathcal{I}_u^-),\ (u, \mathcal{I}_u^+) \succ (u, \mathcal{P}_u),\ (u, \mathcal{P}_u) \succ (u, \mathcal{I}_u^-).$$

We can see that, $\mathcal{P}_u$ is the set of items that u may have interests in returned by the memory-based collaborative filtering.

BPR tries to optimize the standard AUC that is designed for binary classification [3], while we try to optimize a generalized AUC (GAUC) [24, 41],

$$\text{GAUC} = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\Big[\text{AUC}(u, \mathcal{I}_u^+, \mathcal{I}_u^-) + \eta_1\text{AUC}(u, \mathcal{I}_u^+, \mathcal{P}_u) + \eta_2\text{AUC}(u, \mathcal{P}_u, \mathcal{I}_u^-)\Big],$$

where $\text{AUC}(u, \mathcal{A}, \mathcal{B}) = \frac{1}{|\mathcal{A}||\mathcal{B}|}\sum_{i\in\mathcal{A}}\sum_{j\in\mathcal{B}}\delta\left(\hat{\mathbf{R}}_{uij} > 0\right)$ is the standard AUC, η_1 and η_2 are confidence coefficients. We use differentiable loss $\delta(x > 0)$ which is identical to $\ln\sigma(x)$. We can see

that GAUC is the weighted combination of three standard AUC terms. To maximize it, we propose our **S**pectrum-enhanced **P**airwise **L**earning to **R**ank (**SPLR**) optimization:

$$\begin{aligned}\text{SPLR\_OPT} = \sum_{u\in\mathcal{U}}\Bigg[&\sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{I}_u^-}\ln\sigma\left(\hat{\mathbf{R}}_{uij}\right)\\ &+\eta_1\sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{P}_u}\ln\sigma\left(\hat{\mathbf{R}}_{uij}\right)\\ &+\eta_2\sum_{i\in\mathcal{P}_u}\sum_{j\in\mathcal{I}_u^-}\ln\sigma\left(\hat{\mathbf{R}}_{uij}\right)\Bigg]-\frac{\lambda_r}{2}\|\mathbf{\Theta}\|_{\mathrm{F}}^2.\end{aligned}\tag{4}$$

4.2 Model Learning

To maximize the SPLR objective function, we take the first-order derivatives of Equation (4) with respect to each model parameter:

$$\begin{aligned}\nabla_{\mathbf{\Theta}}\text{SPLR\_OPT} = \sum_{u\in\mathcal{U}}\Bigg[&\sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{I}_u^-}\sigma\left(-\hat{\mathbf{R}}_{uij}\right)\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\mathbf{\Theta}}\\ &+\eta_1\sum_{i\in\mathcal{I}_u^+}\sum_{j\in\mathcal{P}_u}\sigma\left(-\hat{\mathbf{R}}_{uij}\right)\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\mathbf{\Theta}}\\ &+\eta_2\sum_{i\in\mathcal{P}_u}\sum_{j\in\mathcal{I}_u^-}\sigma\left(-\hat{\mathbf{R}}_{uij}\right)\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\mathbf{\Theta}}\Bigg]-\lambda_r\,\mathbf{\Theta}.\end{aligned}\tag{5}$$

We use $\boldsymbol{\theta}$ to denote certain row of $\mathbf{\Theta}$, the derivatives in Equation (5) are:

$$\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\boldsymbol{\theta}}=\begin{cases}\mathbf{V}_i-\mathbf{V}_j & \text{if}\quad\boldsymbol{\theta}=\mathbf{U}_u\\ \mathbf{U}_u/-\mathbf{U}_u & \text{if}\quad\boldsymbol{\theta}=\mathbf{V}_i/\mathbf{V}_j\\ \mathbf{F}_i-\mathbf{F}_j & \text{if}\quad\boldsymbol{\theta}=\mathbf{P}_u\\ \mathbf{E}_u/-\mathbf{E}_u & \text{if}\quad\boldsymbol{\theta}=\mathbf{Q}_i/\mathbf{Q}_j\end{cases}.\tag{6}$$

$\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\boldsymbol{\theta}}$ in Equation (6) is certain row of $\frac{\partial\hat{\mathbf{R}}_{uij}}{\partial\mathbf{\Theta}}$ in Equation (5), for example, the u-th row when $\boldsymbol{\theta}=\mathbf{U}_u$.

The detailed procedure is shown in Algorithm 1. We calculate the Laplacian matrices with the Equation (2) (line 1) and decompose them, the first K_1/K_2 eigenvectors of $\mathbf{L}^U/\mathbf{L}^I$ form $\mathbf{E}/\mathbf{F}$ (line 2). We then construct the three sets in Equation (3) for each user (line 3). Lines 5-14 show the process of model learning. For each record $\{u,i\}$, we choose m potential samples $\{j_1,\cdots,j_m\}$ from $\mathcal{P}_u$ (line 9) and m negative samples $\{k_1,\cdots,k_m\}$ from $\mathcal{I}_u^-$ (line 10), m is the sampling rate. The model is optimized with the pairs $\{u,i,j_1\},\cdots,\{u,i,j_m\}$, $\{u,i,k_1\},\cdots,\{u,i,k_m\}$, $\{u,j_1,k_1\},\cdots,\{u,j_m,k_m\}$. We finally calculate the derivatives given by Equation (5) with a batch of pairs (line 12) and update the parameters (line 13).

Like many existing works [30, 33, 49], we also use the connections of users and items to enhance the pairwise learning. However, in previous works, only one-order connections are utilized while in our SPLR model, we leverage connections with all orders. In a hypergraph, we have $\mathbf{L}^n=(\mathbf{I}-\mathbf{A})^n=\sum_{r=1}^{n}C_n^r(-\mathbf{A})^r$, where $\mathbf{A}=\mathbf{D}^{-\frac{1}{2}}\mathbf{H}\mathbf{W}\mathbf{\Delta}^{-1}\mathbf{H}^{\mathsf{T}}\mathbf{D}^{-\frac{1}{2}}$ is the normalized adjacency matrix. $\mathbf{A}^r$ indicates the r-order connections, therefore $\mathbf{L}^n$ contains the information of all connections less than n-order in the hypergraph. For the i-th eigenvector $\mathbf{u}_i$ of $\mathbf{L}$, λ_i is the corresponding eigenvalue, we have $\mathbf{L}\mathbf{u}_i=\lambda_i\mathbf{u}_i$. Since $\mathbf{L}^n\mathbf{u}_i=\lambda_i^n\mathbf{u}_i$, $\mathbf{u}_i$ is also the eigenvector of $\mathbf{L}^n$, therefore $\mathbf{F}$ contains information of all-order connections in the hypergraph. Constructed with the spectral features, latent communities/categories also take the high-order information into consideration, that is the superiority of our model over existing learning to rank methods.

Algorithm 1: Mini-batch gradient descent algorithm

Input: Implicit feedback matrix $\mathbf{R}$, regularization coefficient λ_r, batch size b, learning rate η, weighting parameters η_1 and η_2, sampling rate m, maximum number of iterations $iter\_max$.
Output: top-n predictions given by the completed matrix $\hat{\mathbf{R}}$

1 calculate the Laplacian matrices of user hypergraph $\mathbf{L}^U$ and item hypergraph $\mathbf{L}^I$ from $\mathbf{R}$;
2 decompose the Laplacian matrices to get the spectral features $\mathbf{E}$ and $\mathbf{F}$;
3 cluster vertices and calculate the potential set for each user;
4 initialize $\mathbf{\Theta}$ randomly;
5 **for** $iter = 1$ *to* $iter\_max$ **do**
6 split all purchase records into b-size batches;
7 **for** *each batch* **do**
8 **for** *each record in current batch* **do**
9 select m potential items randomly from $\mathcal{P}_u$;
10 select m negative items randomly from $\mathcal{I}_u^-$;
11 add these items to the current batch;
12 calculate $\nabla_{\mathbf{\Theta}}$SPLR_OPT with current batch;
13 $\mathbf{\Theta}=\mathbf{\Theta}+\eta\nabla_{\mathbf{\Theta}}$SPLR_OPT;
14 calculate $\hat{\mathbf{R}}$ to predict the top-n items;
15 **return** *the top-n items*;

5 EXPERIMENTS

In this section, we design experiments on real-world datasets to validate the effectiveness of our models. We evaluate our proposed methods focusing on the following three key research questions:

RQ1: How is the performance of our entire spectrum-enhanced recommendation model (effectiveness of SCF_SPLR)?
RQ2: How is the performance of the spectral features to model users' preference and items' properties (effectiveness of SCF_BPR)?
RQ3: How is the performance of the SPLR optimization criterion (effectiveness of MF_SPLR)?

Table 1: Statistics of datasets

Dataset	Purchase	User	Item	Sparsity
Clothes	115841	32728	8777	99.9597%
Jewelry	37314	15924	3607	99.9350%

5.1 Datasets

In this paper, we perform experiments on the *Amazon* dataset [14], which is the consumption records from *Amazon.com*. We consider two large categories *Amazon.clothes* and *Amazon.jewelry*, which are filtered from the *Amazon* dataset. We take users' review histories as implicit feedback. Some details of the datasets are shown in Table 1.

Table 2: Recommendation performance (test set)

Datasets	Metrics (%)		MP	PMF	BPR	GBPR	VBPR	SPLR	Improvement BPR	Improvement GBPR
Jewelry	*F*-1@	2	0.808±0.048	3.851±0.144	3.684±0.194	3.811±0.178	3.998±0.165	**4.104**±0.185	11.40%	7.69%
		5	0.747±0.039	3.507±0.120	3.576±0.137	3.656±0.115	3.820±0.068	**3.905**±0.136	9.20%	6.81%
		10	0.608±0.012	3.064±0.082	3.159±0.099	3.192±0.102	**3.354**±0.109	3.339±0.068	5.70%	4.61%
		20	0.425±0.014	2.403±0.026	2.489±0.042	2.515±0.040	**2.653**±0.046	2.638±0.064	5.99%	4.89%
	NDCG@	2	0.618±0.047	3.762±0.128	3.722±0.184	3.845±0.185	3.960±0.192	**4.057**±0.179	9.00%	5.51%
		5	0.513±0.014	2.827±0.075	2.775±0.106	2.924±0.104	3.075±0.102	**3.099**±0.121	11.68%	5.98%
		10	0.402±0.012	2.229±0.047	2.227±0.039	2.301±0.047	2.445±0.043	**2.450**±0.048	10.01%	6.48%
		20	0.280±0.009	1.735±0.039	1.738±0.017	1.788±0.036	**1.892**±0.043	1.861±0.046	7.08%	4.08%
Clothes	*F*-1@	2	0.601±0.043	2.638±0.063	2.831±0.116	2.918±0.107	3.078±0.098	**3.125**±0.087	10.39%	7.09%
		5	0.563±0.032	2.481±0.140	2.689±0.041	2.803±0.045	2.945±0.048	**2.999**±0.070	11.53%	6.99%
		10	0.559±0.021	2.087±0.131	2.322±0.039	2.445±0.040	2.464±0.040	**2.531**±0.063	9.00%	3.52%
		20	0.462±0.027	1.626±0.065	1.828±0.056	1.914±0.054	1.957±0.053	**2.034**±0.022	11.27%	6.27%
	NDCG@	2	0.619±0.032	2.791±0.052	3.026±0.134	3.118±0.126	3.119±0.118	**3.331**±0.144	10.08%	6.83%
		5	0.452±0.039	2.021±0.109	2.185±0.068	2.293±0.054	2.410±0.040	**2.451**±0.028	12.17%	6.89%
		10	0.373±0.009	1.567±0.091	1.740±0.052	1.844±0.047	1.865±0.042	**1.907**±0.042	9.60%	3.42%
		20	0.300±0.007	1.197±0.054	1.330±0.069	1.402±0.045	1.442±0.022	**1.477**±0.020	11.05%	5.35%

5.2 Baselines

We adopt the following methods as baselines for performance comparison:

- **MP:** This **M**ost **P**opular method ranks items according to their popularity. It is a non-personalized method to benchmark the recommendation performances.
- **PMF:** This **P**robabilistic **M**atrix **F**actorization method, which was proposed by [38], is a frequently used state-of-the-art approach for rating-based optimization and prediction. We set the score of positive samples as 1 and missing values as 0.
- **BPR (MF_BPR):** This **B**ayesian **P**ersonalized **R**anking method is the most widely used ranking-based method for implicit feedback [37]. It regards all unobserved samples as negative samples and maximizes the likelihood of users' preference over a pair of positive sample and negative sample.
- **GBPR (MF_GBPR):** This **G**roup Preference-based **B**ayesian **P**ersonalized **R**anking method [30] is an extension of BPR, which tries to relax BPR's assumptions to a group pairwise preference assumption. We fix the number of grouped users to 3.
- **VBPR:** This **V**isual **B**ayesian **P**ersonalized **R**anking method is a stat-of-the-art visual-based recommendation method [14]. The visual features are extracted from the product images with a pre-trained deep convolutional neural network (CNN).

5.3 Experiment Settings

The eigen-decomposition of Laplacian matrices is implemented with `sparse.linalg()` function of `scipy` library in `python`. The *Amazon* dataset is filtered with 5-core (remove users and items with less than 5 purchase records) and records before 2010 is removed. We then filter *Jewelry* and *Clothes* datasets from *Amazon*. Each dataset is split into training set (80%), validation set(10%), and test set (10%) randomly. Cold items and users (items and users with no record in training set) in validation and test sets are removed. We adapt F_1-score and normalized discounted cumulative gain (*NDCG*) to evaluate the performance of the baselines and our proposed model. Mini-batch stochastic gradient descent (MSGD) is leveraged to optimize our model. The learning rate is determined by grid searching in the range of {0.005, 0.01, 0.02, 0.05, 0.1, 0.2} and the batch size is determined in the range of {1000, 2000, 3000, 4000, 5000}. We evaluate different number of latent factors K_0 in the range of {10, 20, 50, 100, 200}, the regularization coefficient λ_r in the range of $\{0, 0.1, 0.2, \cdots, 1.5\}$, and the weighted parameters η_1 and η_2 in the range of $\{0, 0.01, 0.1\}$. We conduct our experiments by predicting top-{2, 5, 10, 20} items to each user. The sampling rate m is set as 5 (select 5 negative/potential samples for each positive sample to construct pairs) in all pairwise learning to balance the accuracy and efficiency.

5.4 Performance of Our Entire Model (RQ1)

In this subsection, we report the performance of our entire model (SCF model optimized with SPLR, SCF_SPLR, abbreviated as SPLR) and baselines to give some analysis. We tune all models in the validation set and test the performance in the test set. The impact of some important hyperparameters, such as the number of latent factors, regularization coefficient, and weighted parameters, are shown. When training, we iterate the whole dataset 200 times to learn models and select 1000 samples randomly from the test/validation set to test all models in each iteration (except MP, we just test 200 times without training). We record the best performance of each model during this procedure as the evaluation of it. We execute all models 5 times and then report the mean and standard deviation. The learning rate is set as 0.05 and the batch size is set as 5000 for all models.

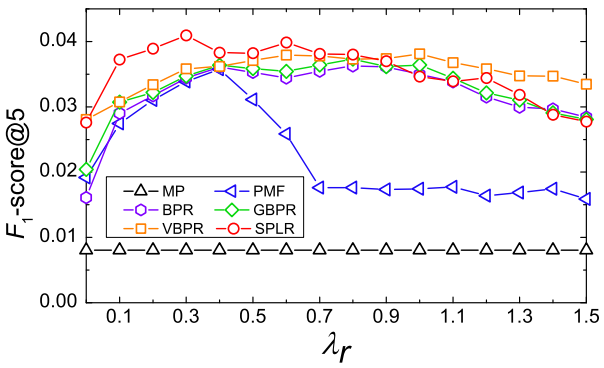
The performance of all models is represented in Table 2, we can see that all personalized models outperform MP significantly in all situations — about 3-6 times improvement. Comparison between BPR and PMF shows the superiority of pairwise learning over point-wise one. Considering the group behaviour, GBPR gets a further enhancement and outperforms BPR in all situations. As an enhanced learning to rank method, we compare our model with BPR and GBPR, and the improvement is shown in the last two columns in Table 2. We can see that our model dramatically outperforms these two models. In GBPR, only one-order user connections are utilized while in SPLR, the spectral features capture high-order connections for both users and items, thus provides high-level and comprehensive description of the similarity among vertices.

We also report the performance of VBPR, the state-of-the-art side information-based model. With the extra visual feature, it performs the best among all baselines. As we can see, our model can even outperform VBPR in most situations. Though both utilizing "side" information to enhance the accuracy, VBPR uses "outside" information, i.e., extra visual data, while SPLR uses "inside" information extracted from the purchase records. From the last two columns of Table 2, we can see that in *Jewelry* dataset, the relative improvement of our model decreases with the increasing of n (the number to recommend), while in *Clothes* dataset, the relative improvement is stable. That is not because the our model performs badly with a large n, it is the property of the dataset. We can see that in *Jewelry* set, the gap (relative improvement) between any two pairwise learning models decreases with the increasing of n.

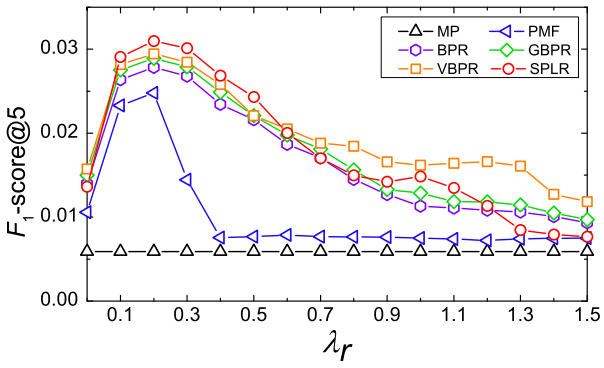
To analyze the sensitivity of our model, we report the performance (F_1-score@5) with different hyperparameters. The impact of regularization coefficient λ_r is represented in Figure 2. Though both filtered from *Amazon* dataset, these two datasets have pretty different properties. From Figure 2(a), we can see that all models perform quiet differently to each other in *Jewelry* dataset, PMF, BPR, GBPR, VBPR, and SPLR get the best performance when λ_r is set to 0.4, 0.8, 0.8, 1.0, and 0.3 respectively. While in Figure 2(b), all models get the best performance when λ_r is set to 0.2. With the increasing of λ_r, accuracy of the point-wise learning method decreases rapidly compared with pairwise learning methods.

We also report F_1-score@5 with different length of latent factors to get the best K_0. In both two datasets, performance of all models (except MP) increase with the increasing of dimensions. We can see that due to the smaller size, the purchase record matrix of *Jewelry* has a smaller rank than that of *Clothes*: The performance in *Jewelry* tends to be stable after K_0 is larger than 50 while the performance in *Clothes* keeps increasing obviously with K_0 changing from 10 to 200. To get the best performance, we set K_0 as 200 for all models.

The impact of weighting parameters is illustrated in Figure 4. Our model performs the best when $\eta_1 = 0.01$ and $\eta_2 = 0.01$ in these two datasets. When $\eta_1 = 0$ and $\eta_2 = 0$, SPLR (SCF_SPLR) degenerates into SCF_BPR. From the figures we can see that SPLR outperforms SCF_BPR 4.70% in *Jewelry* validation dataset and 4.03% in *Jewelry* validation dataset on F_1-score@5. We can see that the enhancement of SLPR is not so significant, this is because we have explored the similarity information by incorporating the spectral features into SCF_BPR model, which outperforms MF_BPR significantly (Table 3). We leverage SPLR optimization to explore the similarity information thoroughly for further accuracy improvement.

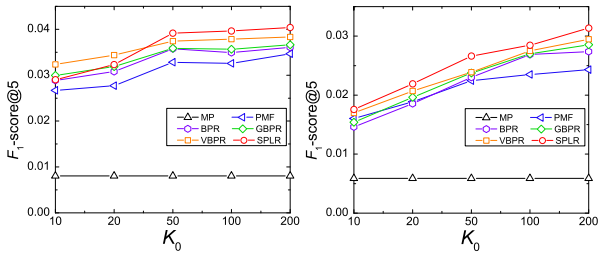


(a) *Jewelry*



(b) *Clothes*

Figure 2: Impact of regularization coefficient λ_r (validation set)



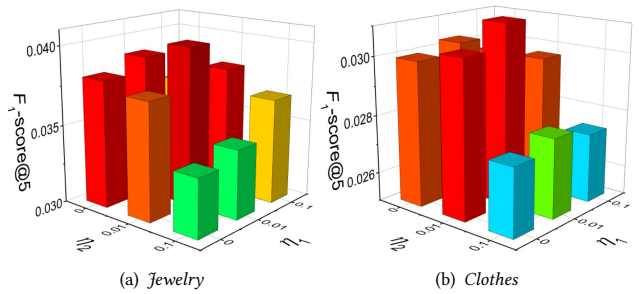
(a) *Jewelry* (b) *Clothes*

Figure 3: Impact of latent factor dimensions K_0 (validation set)

We demonstrated the effectiveness of our entire model in this subsection, and in next subsections, we illustrate the effectiveness of each part of our model — spectral feature-enhanced model (SCF_BPR) and spectral clustering-enhanced pairwise learning optimization strategy (MF_SPLR).

5.5 Effectiveness of SCF_BPR (RQ2)

In this subsection, we denote all models with the model names and optimization methods. We rename BPR as MF_BPR, which means



(a) *Jewelry* (b) *Clothes*

Figure 4: Impact of weighting parameters η_1 and η_2 (validation set)

MF model optimized with BPR algorithm. Similarly, SCF_BPR is SCF model optimized with BPR. We demonstrate the effectiveness of spectral features by comparing SCF_BPR with other baselines. We first report the sensitivity with the number of eigenvectors by grid searching in the range of {0,10,100,1000}. All experiments in this and next subsection are conducted in *Jewelry* dataset.

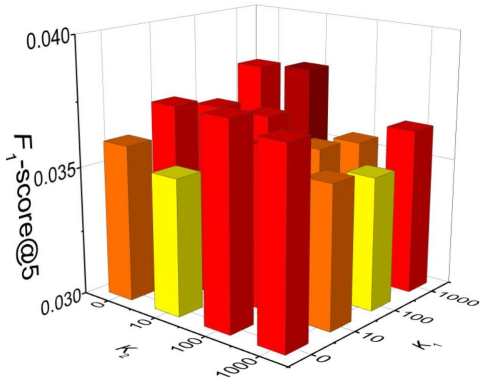


Figure 5: Performance of SCF_BPR with different length of spectral features (*Jewelry*, validation set)

The sensitivity of SCF_BPR with the length of spectral features is shown in Figure 5. When $K_1 = 1000$ and $K_2 = 10$, SCF_BPR performs the best. When $K_1 = 0$ and $K_2 = 0$, SCF_BPR degenerates into MF_BPR. SCF_BPR outperforms MF_BPR 7.2% on F_1-score@5 in *Jewelry* validation set. From Figure 5 we can see that user feature and item feature do not jointly work well since SCF_BPR does not perform well when K_1 and K_2 are both large. In real-world application, we can set $K_1 = 0$ and $K_2 = 100$ to balance the accuracy and the efficiency.

The performance of SCF_BPR and baselines in test set is shown in Table 3. To save space, only the performance of MF_BPR is reported, since it is the most important baseline. We can check Table 2 for the performance of other baselines. With the spectral features providing similarity information, SCF_BPR outperforms MF_BPR 7.27% and 6.38% on F_1-score and *NDCG* respectively in the best situation.

Table 3: Performance comparison for SCF_BPR (*Jewelry*, test set)

Metrics (%)		MF_BPR	SCF_BPR	Improvement
F-1@	2	3.684±0.194	**3.952**±0.221	7.27%
	5	3.576±0.137	**3.768**±0.098	5.37%
	10	3.159±0.099	**3.301**±0.087	4.50%
	20	2.489±0.042	**2.596**±0.029	4.30%
NDCG@	2	3.722±0.184	**3.953**±0.151	6.21%
	5	2.775±0.106	**2.952**±0.074	6.38%
	10	2.227±0.039	**2.366**±0.037	6.24%
	20	1.738±0.017	**1.835**±0.023	5.58%

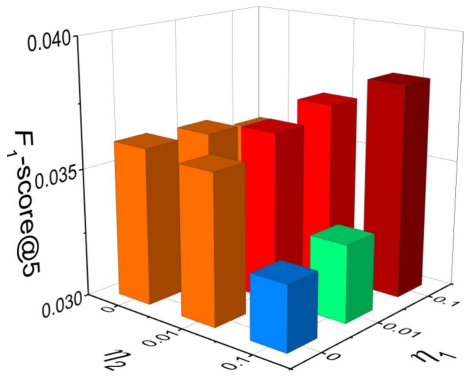


Figure 6: Performance of MF_SPLR with different weighting parameters (*Jewelry*, validation set)

5.6 Effectiveness of MF_SPLR (RQ3)

The sensitivity of MF_SPLR with the weighting parameters η_1 and η_2 are shown in Figure 6. We can see that MF_SPLR performs the best when $\eta_1 = 0.1$ and $\eta_2 = 0.1$. When $\eta_1 = 0$ and $\eta_2 = 0$, MF_SPLR degenerates to MF_BPR. MF_SPLR outperforms MF_BPR 6.13% F_1-score@5 in *Jewelry* validation set. Comparing Figure 4(a) and Figure 6 we can see that though both enhanced with SPLR optimization, MF_SPLR gains more improvement than SCF_SPLR, and the SPLR terms are weighted more in MF_SPLR ($\eta_1 = 0.1$, $\eta_2 = 0.1$ in MF_SPLR while $\eta_1 = 0.01$, $\eta_2 = 0.01$ in SCF_SPLR). It is because that the similarity information has been leveraged in SCF_SPLR by modeling with the spectral features. While in SCF_SPLR, SPLR is the only way to utilize the similarity information.

The performance of MF_SPLR and baselines in *Jewelry* test set is shown in Table 4. From the table we can see that SCF_BPR outperforms MF_BPR 8.20% and 5.42% on F_1-score and *NDCG* respectively in the best situation. Comparing Tables 2, 3, and 4, we can see that the improvement of the entire model is less than the sum of the improvement of each part. That may be because our model utilizes the similarity information by modeling (SCF) and optimizing (SPLR), thus these two parts provide the same information in different ways.

In Figure 7, we show some latent categories of the *Jewelry* set, which contains all watches and jewelries. There are six latent categories in Figure 7. Not like a real category, items in a latent category

Figure 7: Latent categories (*Jewelry* set)

Table 4: Performance comparison for MF_SPLR (*Jewelry*, test set)

Metrics (%)		MF_BPR	MF_SPLR	Improvement
F-1@	2	3.684±0.194	**3.986**±0.074	8.20%
	5	3.576±0.137	**3.789**±0.123	5.96%
	10	3.159±0.099	**3.314**±0.056	4.91%
	20	2.489±0.042	**2.578**±0.054	3.58%
NDCG@	2	3.722±0.184	**3.924**±0.151	5.43%
	5	2.775±0.106	**2.923**±0.074	5.33%
	10	2.227±0.039	**2.339**±0.037	5.03%
	20	1.738±0.017	**1.797**±0.023	3.39%

are relevant items or similar items (items strongly connected in the hypergraph). For example, in Figure 7(a), items are watches and some relevant commodities, such as watch bands and watch boxes[2]. Users who want to by watches may have interests in this category. Though items in the same latent categories may be different kinds of commodities, they are in the same style. For example, items in Figure 7(a) are designed for business men, they are luxurious, with metallic luster, look mature and steady. On the contrary, different categories may contain items of the same kind, but they are in different styles. For example, the category in Figure 7(b) is also for watches, however is in a totally different style from that in Figure 7(a). Most items in this category are plastic and cheap digital watches for sports and boys may have interests in them. Items in Figure 7(c) are watches and jewelries for young girls, they look simple and elegant. Jewelries in Figure 7(d) and Figure 7(e) are luxurious and exaggerative, which are full of diamonds and gemstones. Items in Figure 7(f) are in punk style and rebellious teenagers may prefer them.

[2]https://www.amazon.com/dp/B005IHDLYC/?tag=tc0f3f-20

Table 5: Performance with different features (*Jewelry*, test set)

Metrics (%) \ Features		None	Spectral features	CNN feature	Spectral features & CNN feature
F-1@	2	3.684±0.194	4.104±0.185	4.218±0.213	**4.414**±0.177
	5	3.576±0.137	3.905±0.136	3.974±0.106	**4.108**±0.165
	10	3.159±0.099	3.339±0.068	3.398±0.103	**3.567**±0.120
	20	2.489±0.042	2.638±0.064	2.631±0.067	**2.800**±0.103
NDCG@	2	3.722±0.184	4.057±0.179	4.184±0.165	**4.392**±0.278
	5	2.775±0.106	3.099±0.121	3.173±0.090	**3.315**±0.183
	10	2.227±0.039	2.450±0.048	2.542±0.078	**2.623**±0.137
	20	1.738±0.017	1.861±0.046	1.934±0.073	**2.011**±0.092

Figure 7 shows how the item spectral feature helps to recommend. The item spectral feature indicates the style of the item, circle of interest the item belongs to, the price level, target users, etc. Modeling with the item spectral feature (SCF) makes the items in the same latent category tend to have similar score. And optimizing with the item spectral feature (SPLR) makes the items in the positive category (latent category containing positive samples) tend to have a high score. For the user spectral feature, we can draw the same conclusion.

Both spectral features and latent factors are extracted from graph structures which are constructed from the purchase records (one bipartite graph and two hypergraphs). In fact, they both mine the similarity information while with different emphasis — user/item spectral features describe the similarity of users/items, while latent factors describe the similarity between a user and an item. Spectral features and latent factors complete each other in describing the similarity information and enhance the effectiveness of the recommendation model.

6 EXTENSION

In this paper, we proposed new spectral features and incorporated them into an MF model to predict users' preference. We then introduced new spectral clustering-based pairwise learning method to optimize our model. In this section, we extend our model and propose a framework of learning to rank models enhanced with side information features. Not only the spectral features, all features can be leveraged in our framework. Assuming there are n features for users $\mathbf{E}^{(1)}, \cdots, \mathbf{E}^{(n)}$ and m features for items $\mathbf{F}^{(1)}, \cdots, \mathbf{F}^{(m)}$, the prediction is given by:

$$\hat{\mathbf{R}} = \mathbf{U}\mathbf{V}^{\mathsf{T}} + \sum_{j=1}^{m} \mathbf{P}^{(j)}\mathbf{F}^{(j)\mathsf{T}} + \sum_{k=1}^{n} \mathbf{E}^{(k)}\mathbf{Q}^{(k)\mathsf{T}},$$

where $\mathbf{P}_u^{(j)}$ is the preference of user u based on the j-th item feature. These item features can be high-level features like the spectral feature, CNN feature, auditory feature, or low-level features like the color histogram or tags. $\mathbf{Q}_i^{(k)}$ is the fitness of item i based on the k-th user feature. These user features can be high-level features like the spectral feature, or the low-level information vector containing such as age, genders, addresses, etc.

For pairwise learning, we use the latent community $C_u = \bigcup_{j=1}^{n} C_u^{(j)}$ and latent category $C_i = \bigcup_{k=1}^{m} C_i^{(k)}$ to construct the potential set in Equation (3), where $C_u^{(j)}$ is the latent community of user u clustered by the j-th feature and $C_i^{(k)}$ is the latent category of item i clustered by the k-th feature.

Experimental results of our extended model are shown in Table 5. These four columns are performances of our extended models with no feature (i.e., MF_BPR), with the spectral features (i.e., SCF_SPLR), with the CNN feature (i.e., VBPR optimized with visual clustering-enhanced pairwise learning to rank), and with the spectral features plus the CNN feature. Enhanced with the spectral features, SCF_SPLR outperforms MF_BPR 9.20% on F_1-score@5. Leveraging appearance information, our extended model with the CNN feature outperforms MF_BPR 11.13% on F_1-score@5. Comparing Table 2 and Tale 5 we can see that it also outperforms VBPR 4.03% on F_1-score@5 due to the enhanced pairwise learning optimization. Our extended model with the CNN feature and spectral features outperforms MF_BPR 14.88% on F_1-score@5.

7 CONCLUSION AND FUTURE WORK

In this paper, we investigated the usefulness of the spectral features in recommendation tasks. We first introduced novel spectral features, which contain similarity information, and injected them into an MF structure to model users' preference and items' properties. We then clustered the spectral features to construct the latent communities and categories for users and items respectively, and used them to enhance the pairwise learning. We finally extended our model and proposed a framework for side information-enhanced pairwise learning. Experiments on challenging real-world datasets show that our proposed methods significantly outperform state-of-the-art models.

For future work, we will investigate the effectiveness of our proposed spectral features in the setting of explicit feedback. Also, we are interested in explaining the recommendation result to users with the information of latent communities and categories. Lastly, we will use neural networks [16, 18] to learn how to combine features, such as the item spectral feature and user spectral feature, or spectral features and other kinds of features.

REFERENCES

[1] James Bennett and Stan Lanning. 2007. The netflix prize. In *KDDCup*. New York, NY, USA, 35.

[2] Rianne Van Den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph Convolutional Matrix Completion. *Computing Research Repository - CoRR* (2017).

[3] Andrew P. Bradley. 1997. The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms. *Pattern Recogn.* (1997), 1145–1159.

[4] D. Cai, X. He, X. Wu, and J. Han. 2008. Non-negative Matrix Factorization on Manifold. In *2008 Eighth IEEE International Conference on Data Mining (ICDM '08)*. 63–72.

[5] Zhe Cao, Tao Qin, Tie Yan Liu, Ming Feng Tsai, Dawen Li, Hang Liang, Minshu Zhan, and Daniel P. W. Ellis. 2015. Content-Aware Collaborative Music Recommendation Using Pre-trained Neural Networks. In *International Society for Music Information Retrieval (ISMIR '15)*. 129–136.

[6] Zhe Cao, Tao Qin, Tie Yan Liu, Ming Feng Tsai, and Hang Li. 2007. Learning to rank:from pairwise approach to listwise approach. In *International Conference on Machine Learning (ICML '07)*. 129–136.

[7] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive Collaborative Filtering: Multimedia Recommendation with Item- and Component-Level Attention. In *International Conference on Research and Development in Information Retrieval (SIGIR '17)*. 335–344.

[8] Tao Chen, Xiangnan He, and Min-Yen Kan. 2016. Context-aware Image Tweet Modelling and Recommendation. In *ACM Multimedia (MM '16)*. 1018–1027.

[9] Xu Chen, Yongfeng Zhang, Qingyao Ai, Hongteng Xu, Junchi Yan, and Zheng Qin. 2017. Personalized Key Frame Recommendation. In *International Conference on Research and Development in Information Retrieval (SIGIR '17)*. 315–324.

[10] Jingtao Ding, Fuli Feng, Xiangnan He, Guanghui Yu, Yong Li, and Depeng Jin. 2018. An Improved Sampler for Bayesian Personalized Ranking by Leveraging View Data. In *International World Wide Web Conference (WWW '18)*. 13–14.

[11] Fuli Feng, Xiangnan He, Yiqun Liu, Liqiang Nie, and Tat Seng Chua. 2018. Learning on Partial-Order Hypergraphs. In *International World Wide Web Conference (WWW '18)*. 1523–1532.

[12] David Goldberg. 1992. Using collaborative filtering to weave an information tapestry. *Communications of the Acm* (1992), 61–70.

[13] Roger G. Grimes, John G. Lewis, and Horst D. Simon. 1994. A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems. *Siam Journal on Matrix Analysis & Applications* (1994), 228–272.

[14] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In *AAAI Conference on Artificial Intelligence (AAAI '16)*. 144–150.

[15] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *International Conference on Research and Development in Information Retrieval (SIGIR '17)*. 355–364.

[16] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. 2018. Outer Product-based Neural Collaborative Filtering. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI '18)*. 2227–2233.

[17] Xiangnan He, Zhenkui He, Jingkuan Song, Zhenguang Liu, Yu Gang Jiang, and Tat Seng Chua. 2018. NAIS: Neural Attentive Item Similarity Model for Recommendation. *IEEE Transactions on Knowledge & Data Engineering* (2018), 1–1.

[18] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *International World Wide Web Conference (WWW '17)*. 173–182.

[19] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *International Conference on Research and Development in Information Retrieval (SIGIR '16)*. 549–558.

[20] W. Hwang, J. Parc, S. Kim, J. Lee, and D. Lee. 2016. âĂIJTold you i didn't like itâĂİ: Exploiting uninteresting items for effective collaborative filtering. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE '16)*. 349–360.

[21] Mohsen Jamali and Martin Ester. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *ACM Conference on Recommender Systems (RecSys '10)*. 135–142.

[22] Yehuda Koren. 2009. The bellkor solution to the netflix grand prize. *Netflix prize documentation* (2009), 1–10.

[23] Hadrien Van Lierde and Tommy W. S. Chow. 2017. A Hypergraph Model for Incorporating Social Interactions in Collaborative Filtering. In *International Conference On Data Mining, Communications And Information Technology (DMCIT '17)*. 32.

[24] Hongzhi Liu, Zhonghai Wu, and Xing Zhang. 2018. CPLR: Collaborative Pairwise Learning to Rank for Personalized Recommendation. *Knowledge-Based Systems* (2018).

[25] Juntao Liu, Caihua Wu, Yi Xiong, and Wenyu Liu. 2014. List-wise probabilistic matrix factorization for recommendation. *Information Sciences* (2014), 434–447.

[26] Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *ACM Conference on Recommender Systems (RecSys '13)*. 165–172.

[27] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *International Conference on Research and Development in Information Retrieval (SIGIR '15)*. 43–52.

[28] Narang, K Sunil, Ortega, Antonio, Vandergheynst, and Pierre. 2013. The Emerging Field of Signal Processing on Graphs. *IEEE Signal Processing Magazine* (2013), 83–98.

[29] Andrew Y Ng, Michael I Jordan, and Yair Weiss. 2002. On spectral clustering: Analysis and an algorithm. In *Annual Conference on Neural Information Processing Systems (NIPS '02)*. 849–856.

[30] Weike Pan and Li Chen. 2013. GBPR: group preference based Bayesian personalized ranking for one-class collaborative filtering. In *International Joint Conference on Artificial Intelligence (IJCAI '13)*. 2691–2697.

[31] Weike Pan, Hao Zhong, Congfu Xu, and Zhong Ming. 2015. Adaptive Bayesian Personalized Ranking for Heterogeneous Implicit Feedbacks. *Know.-Based Syst.* (2015), 173–180.

[32] Johan Paratte and Lionel Martin. 2016. Fast Eigenspace Approximation using Random Signals. *Computing Research Repository - CoRR* (2016).

[33] Shuang Qiu, Jian Cheng, Ting Yuan, Cong Leng, and Hanqing Lu. 2014. Item Group Based Pairwise Preference Learning for Personalized Ranking. In *International Conference on Research and Development in Information Retrieval (SIGIR '14)*. 1219–1222.

[34] Nikhil Rao, Hsiang-Fu Yu, Pradeep K Ravikumar, and Inderjit S Dhillon. 2015. Collaborative Filtering with Graph Information: Consistency and Scalable Methods. In *Annual Conference on Neural Information Processing Systems (NIPS '15)*. 2107–2115.

[35] Steffen Rendle. 2011. Factorization Machines. In *International Conference on Data Mining (ICDM '11)*. 995–1000.

[36] Steffen Rendle and Christoph Freudenthaler. 2014. Improving Pairwise Learning for Item Recommendation from Implicit Feedback. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM '14)*. 273–282.

[37] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Conference on Uncertainty in Artificial Intelligence (UAI '09)*. 452–461.

[38] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *Annual Conference on Neural Information Processing Systems (NIPS '07)*. 1257–1264.

[39] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *International World Wide Web Conference (WWW '01)*. 285–295.

[40] M. Sharir. 1981. A strong-connectivity algorithm and its applications in data flow analysis. *Computers & Mathematics with Applications* (1981), 67–72.

[41] Dongjin Song and David A Meyer. 2015. Recommending Positive Links in Signed Social Networks by Optimizing a Generalized AUC.. In *AAAI Conference on Artificial Intelligence (AAAI '15)*. 290–296.

[42] Frank Edward Walter, Stefano Battiston, and Frank Schweitzer. 2008. A Model of a Trust-based Recommendation System on a Social Network. *Autonomous Agents and Multi-Agent Systems* (2008), 57–74.

[43] Mao Ye, Peifeng Yin, Wang Chien Lee, and Dik Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *International Conference on Research and Development in Information Retrieval (SIGIR '11)*. 325–334.

[44] Wenhui Yu, Huidi Zhang, Xiangnan He, Xu Chen, Li Xiong, and Zheng Qin. 2018. Aesthetic-based Clothing Recommendation. In *International World Wide Web Conference (WWW '18)*. 649–658.

[45] Quan Yuan, Gao Cong, Zongyang Ma, Aixin Sun, and Nadia Magnenat Thalmann. 2013. Time-aware point-of-interest recommendation. In *International Conference on Research and Development in Information Retrieval (SIGIR '13)*. 363–372.

[46] Hanwang Zhang, Fumin Shen, Wei Liu, Xiangnan He, Huanbo Luan, and Tat Seng Chua. 2016. Discrete Collaborative Filtering. In *International Conference on Research and Development in Information Retrieval (SIGIR '16)*. 325–334.

[47] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. 2013. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *International Conference on Research and Development in Information Retrieval (SIGIR '13)*. 785–788.

[48] Lili Zhao, Zhongqi Lu, Sinno Jialin Pan, and Qiang Yang. 2016. Matrix Factorization+ for Movie Recommendation. In *International Joint Conference on Artificial Intelligence (IJCAI '16)*. 3945–3951.

[49] Tong Zhao, Julian McAuley, and Irwin King. 2014. Leveraging Social Connections to Improve Personalized Ranking for Collaborative Filtering. In *ACM International Conference on Information and Knowledge Management (CIKM '14)*. 261–270.

[50] Lei Zheng, Chun Ta Lu, Fei Jiang, Jiawei Zhang, and Philip S. Yu. 2018. Spectral Collaborative Filtering. In *Proceedings of the 12th ACM Conference on Recommender Systems (WSDM '18)*. 311–319.

[51] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. 2006. Learning with Hypergraphs: Clustering, Classification, and Embedding. In *Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems (NIPS '06)*. 1601–1608.