Noname manuscript No.
(will be inserted by the editor)

HHMF: Hidden Hierarchical Matrix Factorization for Recommender Systems

Hui Li[1] · Yu Liu[2] · Yuqiu Qian[2] · Nikos Mamoulis[3] · Wenting Tu[4] · David W. Cheung[2]

Received: date / Accepted: date

Abstract Matrix factorization (MF) is one of the most powerful techniques used in recommender systems. MF models the (user, item) interactions behind historical explicit or implicit ratings. Standard MF does not capture the hierarchical structural correlations, such as publisher and advertiser in advertisement recommender systems, or the taxonomy (e.g., tracks, albums, artists, genres) in music recommender systems. There are a few hierarchical MF approaches, but they require the hierarchical structures to be known beforehand. In this paper, we propose a *Hidden Hierarchical Matrix Factorization* (HHMF) technique, which learns the *hidden* hierarchical structure from the user-item rating records. HHMF does not require the prior knowledge of hierarchical structure; hence, as opposed to existing hierarchical MF methods, HHMF can be applied when this information is either explicit or implicit. According to our

Hui Li
E-mail: hui@xmu.edu.cn

Yu Liu, Yuqiu Qian, David W. Cheung
E-mail: {yliu4, yqqian, dcheung}@cs.hku.hk

Nikos Mamoulis
E-mail: nikos@cs.uoi.gr

Wenting Tu (corresponding author)
E-mail: tu.wenting@mail.shufe.edu.cn

[1] Fujian Key Laboratory of Sensing and Computing for Smart City, School of Information Science and Engineering, Xiamen University, Xiamen, Fujian, China

[2] Department of Computer Science, The University of Hong Kong, Hong Kong

[3] Department of Computer Science and Engineering, University of Ioannina, Ioannina, Epirus, Greece

[4] School of Information Management and Engineering, Shanghai University of Finance and Economics, Shanghai, China

extensive experiments, HHMF outperforms existing methods, demonstrating that the discovery of latent hierarchical structures indeed improves the quality of recommendation.

1 Introduction

Recommender systems have become standard add-ons in popular e-commerce applications and social media sites. Real applications include product recommendation (Amazon[1]), movie recommendation (Netflix[2]), restaurant recommendation (Yelp[3]), and trip recommendation (TripAdvisor[4]). The fundamental task in modern recommender systems is *rating prediction*. Specifically, consider a set of users and a set of items: each user can rate any item by giving it a score either explicitly (e.g., by rating it with a number of stars) or implicitly (e.g., browsing to the web page of the item indicates the user's interest). Given a target user, for each item that he/she has not rated, the system predicts the user's rating (i.e., interest), based on the historical ratings by him/her and other users. Then, the unrated items with the highest predicted ratings are offered as suggestions to the target user.

Matrix factorization (MF) is the most prevalent method for rating prediction in recommender systems, due to its ability to deal with large user-item rating matrices (Li 2018). MF has shown its power in various open competitions (e.g., Netflix prize challenge (Bell and Koren 2007) and KDD Cup 2011 (Dror et al. 2012)) and industrial applications (Koren et al. 2009). We observe that users and items in many applications have structural correlations. Specifically, users may form groups (e.g., a reading group with a specific topic in the book recommender system *goodreads*[5]) and items may be divided into categories (e.g., neighborhoods of attractions recommended by TripAdvisor). MF itself may not be able to capture such structural information and use it for rating prediction. Hence, some extensions of standard MF approaches use side-information to overcome this limitation. For example, MF has been extended to consider social network features such as communities (Li et al. 2015; Li 2015) and trust circles (Yang et al. 2012), where a community or a trust circle consists of strongly connected users. In addition, item taxonomies have also been considered in many MF based approaches, since items from the same group may be similar (e.g., songs belonging to the same album share characteristics) (Koenigstein et al. 2011). These approaches show that incorporating group structural information into MF can further improve the accuracy of rating prediction.

While most of the previous works only consider one-level structures, data in many real recommender systems contain multi-level structures. For exam-

[1] http://www.amazon.com
[2] http://www.netflix.com
[3] http://www.yelp.com
[4] http://www.tripadvisor.com
[5] http://www.goodreads.com

ple, in *advertisement recommender systems* (Oentaryo et al. 2014), a page is associated with a publisher and each publisher belongs to a unique channel. Advertisers provide ads and agree on commissions for customer actions (e.g., clicking an ad), while publishers display ads on some web pages and earn commissions based on the traffic driven to the advertisers. Figure 1 illustrates the hierarchical structure of such a system, which aims at maximizing the revenue based on this multi-level structure. Another example is the Yahoo! Music recommender system, where a four-level taxonomy (tracks, albums, artists, genres) exists. The system leverages such hierarchical information to predict the song that users may like (Koenigstein et al. 2011). To improve the service of multi-level recommender systems, several *hierarchical matrix factorization* methods (Shan et al. 2012; Zhong et al. 2012; Wang et al. 2014) have been proposed.

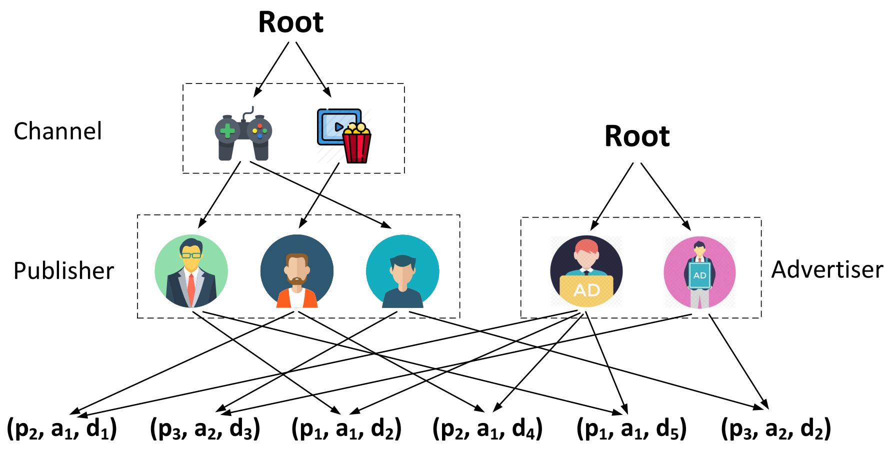


Fig. 1 Multi-level advertisement recommender system where each *page-ad-day* triple (p_i, a_j, d_k) represents that ad from advertiser j on the web page from publisher i was clicked on day d_k.

In this paper, we propose a novel hierarchical MF method called *Hidden Hierarchical Matrix Factorization* (HHMF), which can capture the *hidden* hierarchical structure in the user-item rating records. Unlike traditional MF methods, HHMF takes hierarchical information into consideration. Compared with the few existing hierarchical MF methods, HHMF does not require any prior knowledge of hierarchical structures and thus can be used by both the systems with explicit hierarchical structures and the systems wherein such structures are implicit. Our experimental results show that HHMF outperforms existing methods, demonstrating that the discovery of latent hierarchical structures indeed improves rating prediction accuracy. What is more, our technique is orthogonal to context-based approaches, and its performance is expected to be better when additional information (e.g., user-generated text (Wang et al.

2018a; García-Durán et al. 2018), check-in records (Lu et al. 2017), item meta-data (Li et al. 2012), user grouping data (Ding et al. 2017; Li et al. 2015), relationships in a graph (Qian et al. 2017) or time series data (Li et al. 2019)) is taken into consideration. In summary, the main contributions of this paper are as follows:

- **Upstream hidden structure learning.** Our proposed HHMF approach learns the hidden hierarchical structure for users and items from the rating records, employing a bottom-up iterative optimization method.
- **Downstream rating prediction.** HHMF adopts a top-bottom mechanism for rating prediction, using the grouping results from the upstream phase.
- **Analysis of time complexity.** We conduct an analysis regarding the time complexity of HHMF and show that it has the same complexity as existing hierarchical MF methods, while HHMF is significantly more effective than these methods.
- **Validation.** We conduct comprehensive experiments on several real datasets to validate the effectiveness and efficiency of the proposed method. According to the results, HHMF outperforms existing state-of-the-art methods.

Currently, there are a few limitations of HHMF (see Section 5 for a detailed discussion). Firstly, HHMF only considers basic user-item rating information, though it is possible to incorporate additional auxiliary information. Moreover, the number of layers is a user-defined hyperparameter in HHMF and HHMF adopts *hard-assignment*, i.e., one user group or item group can only be affiliated to one topic group. We will address these limitations in our future work.

The rest of this paper is organized as follows: Section 2 provides background and discusses related work. We present the two phases of HHMF, i.e., upstream hidden hierarchical structure learning and downstream rating prediction, in Section 3. Experiments on real datasets that demonstrate the practicality and effectiveness of HHMF are presented in Section 4. Section 5 concludes the paper and discusses directions for future work.

2 Preliminaries

2.1 Matrix factorization

Let R be an $m \times n$ matrix with the ratings of m users on n items. Basic matrix factorization predicts the missing ratings in R by approximating R as the product of a a d-rank user-specific matrix $Q \in \mathbb{R}^{m\times d}$ with a d-rank item-specific matrix $P \in \mathbb{R}^{n\times d}$, i.e., $R \approx \hat{R} = QP^T$, where $d \ll \min\{m, n\}$. The d-dimensional user vector $Q_u \in Q$ and item vector $P_i \in P$ can be thought of as the latent user preferences and item properties. The inner product between Q_u and P_i is used to model the degree of the match between user u and item

i. A larger inner product implies a higher chance of the user being satisfied by the item. Given a training corpus of ratings R, Q and P are obtained through minimizing the following loss function (Mean Squared Error):

$$f(R|\Theta) = \sum_{r_{ui} \in R} (r_{ui} - \langle Q_u, P_i \rangle)^2 + \lambda(||Q||^2 + ||P||^2), \tag{1}$$

where $\|\cdot\|^2$ denotes the Frobenius norm, Θ represents the parameters and $\langle Q_u, P_i \rangle$ is the dot product of Q_u and P_i. Methods based on gradient descent or alternating least-squares can be used to optimize Q and P (Koren et al. 2009).

The power of MF has been proved in the Netflix Prize competition.[6] MF scales easily to millions or billions of users and items (Facebook 2015; Li et al. 2017). In addition to its superior performance, another strength of MF, making it widely used, is that additional information besides the existing ratings can be integrated into the model to further increase its accuracy. Such information includes social network data (Ma et al. 2011; Li et al. 2015), locations of users and items (Lian et al. 2014; Lu et al. 2017), visual appearance (He and McAuley 2016) and review text (Wang et al. 2018a; García-Durán et al. 2018).

2.2 Related Work

In this section we review MF based recommender approaches, dividing them to *single-layer MF* methods that do not use hierarchical information and *hierarchical MF* techniques, which use this information.

2.2.1 Single-Layer Matrix Factorization

There are four types of correlations from individuals to communities: (user, item), (user, item group), (user group, item) and (user group, item group) in recommender systems (Wang et al. 2014). Basic matrix factorization (Koren et al. 2009) only considers *one-to-one* correlation (i.e., one user to one item). Readers can refer to (Shi et al. 2014) for a detailed introduction and literature review for standard MF, based on the *one-to-one* correlation. Additionally, there are many approaches extending MF to leverage other correlations:

• **Many-to-One Correlation (User $\rightarrow$ Item Groups).** This correlation is common in recommender systems where items form a general-to-specific hierarchy. Mashhoori and Hashemi (2012) propose a MF-based method utilizing this; the latent factor vector of the parent item group is added to Equation 1. Koenigstein et al. (2011) leverage the information from the four-level taxonomy in Yahoo! Music (i.e., tracks, albums, artists and genres) and add taxonomy bias terms to Equation 1 in order to improve the prediction accuracy. The model is enhanced by letting item biases share components for items linked in the taxonomy. Wang and Blei (2011); Wang et al. (2012) integrate latent

[6] http://www.netflixprize.com

Dirichlet allocation (LDA) (Blei et al. 2003) into MF, which tries to use a class of items with the same topic instead of individual items, to improve the quality of recommendation.

• **Many-to-One Correlation (Item $\rightarrow$ User Groups).** This correlation is common when users form social or trust networks.[7] Based on the rationale that a user's interest is similar to or influenced by the user's neighbors in the network, MF has been extended to a network-based model. Users in the network tend to establish relationships with people who share similar interests with them. With this observation, community-based MF (Li et al. 2015) and trust circle based MF (Yang et al. 2012) are proposed. In these two approaches, social constraints are added to Equation 1 so that the latent vectors of users who belong to the same group are similar. Besides, there are some methods in this category which can discover social structure information and make rating predictions at the same time. The user latent factor vectors in Sorec (Ma et al. 2008) are learned based on both the user-item rating matrix and the user-user adjacency matrix. Though Ma et al. (2008) have not systematically studied the task of structure discovery, Sorec can be used for link prediction among users in a social network. Jamali et al. (2011) propose GSBM, which is an extension of the mixed membership stochastic blockmodel (Airoldi et al. 2008). GSBM can make rating predictions and learn the group membership assignments for both users and items in the social network.

• **Many-to-Many Correlation (User/Item Groups $\longleftrightarrow$ User/Item Groups).** CMR (Xu et al. 2014) harnesses reviews associated with ratings and apply co-clustering (Dhillon 2001) to uncover hidden user communities and hidden item groups in social recommender systems, in order to further improve the accuracy of rating predictions. George and Merugu (2005) also use co-clustering to group users and items with similar contexts. Their method then incorporates the biases of users, items and clusters into the rating matrix. The optimization is performed over the new matrix instead of the original rating matrix. Instead of co-clustering, several context-aware recommender systems (Liu and Aberer 2013; Zhong et al. 2012; Wang et al. 2016) use random decision trees (Fan et al. 2003) to group the ratings with similar contexts, but they can only handle categorical contexts. As a comparison, Chen et al. (2014) use spectral clustering (Ng et al. 2001) for user-item subgrouping so that the model can handle both categorical and continuous contexts. Note that previous work in this category still focuses on the prediction of how well one user matches one item during the process, though the information from groups is taken into consideration.

In summary, most of the previous work for single-layered MF either only considers one of the four correlations or requires additional contextual information beyond the user-item rating matrix. As a comparison, our HHMF

[7] Users in a trust network can indicate whether he/she thinks one review with a rating is 'useful'. One trust statement forms an edge with the trust rating as edge weight between two users. Epinions (`http://www.epinions.com`) is an exemplified recommender systems based on trust network.

approach considers and predicts all four correlations during the factorization and it does not require any additional contextual data.

2.2.2 Hierarchical Matrix Factorization

As illustrated in Section 1, structural characteristics in many recommender systems are multi-level. Recently, some hierarchical MF approaches have been proposed to capture the multi-level information. HPMF (Shan et al. 2012) is a hierarchical probabilistic MF method for trait prediction tasks in the plant kingdom. Although HPMF is designed for trait data, it could be generalized and applied to recommender systems. However, HPMF assumes that the data matrices at all levels are available and it only combines two types of correlations, i.e., (user, item) and (user group, item). RPMF (Zhong et al. 2012) applies the idea of random decision trees in the MF framework. It divides the rating matrix iteratively into multi-level rating matrices using random partitioning. In each sub-matrix, RPMF still focuses on the prediction of one user on one item and standard MF is performed to give the prediction. Menon et al. (2011) use hierarchical regularization in MF to improve response prediction in online advertising. Priors based on the hierarchical structure are used as regularization to induce correlations among different levels in the hierarchy. Oentaryo et al. (2014) propose a *Hierarchical Importance-aware Factorization Machine* (HIFM). Since pages and ads can be organized into predefined hierarchies as illustrated in Figure 1, their models incorporate hierarchical information to alleviate the cold-start pages and ads (i.e., pages with few views and ads with few clicks can benefit from the information of their siblings in hierarchies).

However, all aforementioned approaches require the prior knowledge of hierarchies. To overcome this limitation, HGMF (Wang et al. 2014) considers the correlations not only between users and items but also between user groups and item groups. HGMF also provides a greedy clustering algorithm to obtain hierarchical structures for systems where such information is not available. Although HGMF considers all the four correlations introduced in Section 2.2.1, its clustering method is independent of its factorization process. As a comparison, our HHMF approach can make rating predictions and discover structures at the same time; these two tasks benefit each other as the iterative process goes on. IHSR (Wang et al. 2015, 2018b) is another framework which can capture implicit hierarchical structures of users and items. IHSR is based on weighted nonnegative matrix factorization (WNMF) (Zhang et al. 2006). IHSR recursively performs nonnegative matrix factorization (NMF) on the user preference matrix Q and the item characteristic matrix P: $Q \approx \tilde{Q}\dot{Q}$, $P \approx \tilde{P}\dot{P}$. Due to the nonnegativity of $\tilde{Q}$ and $\tilde{P}$, IHSR can use them to indicate the affiliation of a user/item to different groups. $\dot{Q}$ and $\dot{P}$ are the latent matrices of the corresponding user/item groups, which can be further decomposed to generate a deeper hierarchy. IHSR adopts the ideas of identifying communities of users (Wang et al. 2011) and document clustering (Xu et al. 2003), where the affiliations of users/items to groups are represented by nonnegative values.

In summary, IHSR relies on NMF, while any MF method can be used as the basic MF model in HHMF and HHMF is more general than IHSR.

Recently, there are a few approaches combining MF and neural networks (He et al. 2017; Xue et al. 2017). Although these models are represented as hierarchical structures due to the adaptation of neural networks, such hierarchical structures do not reflect the hierarchical user/item grouping like HGMF and HHMF. Instead, each neural layer in these models projects a latent factor vector to another vector in order to finally push the model towards local optima. As we will show in Section 4, HHMF, which appropriately models user/item hierarchical grouping, exhibits better results than neural network-based methods.

2.2.3 Recommendation Models Beyond Matrix Factorization

In addition to MF based methods, there are other recommendation models which consider the affiliations of users and items to groups (Maleszka et al. 2013; He et al. 2016; Nikolakopoulos et al. 2015). For instance, CoBaFi (Beutel et al. 2014) uses bi-clustering (i.e., mixture of Gaussians) to allow for dynamic allocation of statistical capacity between sets of users in a single-layer manner. The motivation behind it is that users with a significant number of ratings can be modeled using *personal* parameter vectors, while cold-start users with little data are probably best modeled as unspecific members of a large pool of similar participants. In this paper, we focus on MF based methods, because MF is the most prevalent method used in recommender systems; hence, we do not discuss more about non-MF methods here.

3 Hidden Hierarchical Matrix Factorization

In this section, we introduce our *Hidden Hierarchical Matrix Factorization* (HHMF) approach. The most frequently used notations are summarized in Table 1. Since we focus on *groups* of users/items in this paper, we modify the traditional notation for MF approaches as used in Section 2.1. From this point on in this paper, u/i will indicate a group of users/items, even for the case where the group only has one user/item. Superscript '(ℓ)' indicates the hierarchal level. For example, $u_1^{(1)}$ indicates user group 1 at level 1 (i.e., it contains only one user u_1 as in flat MF), $u_2^{(2)}$ means the second user group at level 2 which may contain more than one users, $Q^{(\ell)}$ is the user latent vectors at level ℓ and $\theta^{(\ell)}$ is the user topic distribution at level ℓ. We use subscript with a single letter to indicate a vector, e.g., Q_u is the latent factor vector of user group u and θ_u is the topic distribution of user group u. To denote an entry in the vector, we normally use subscript with two letters, e.g., $\theta_{(u,z)}$ is the affiliation of user group u to topic group z and g_{ui} is the topic assignment of user group u on item group i. Other notations are described where they are used.

Table 1 Notations used in HHMF

Symbol	Description
d	Number of latent dimensions/topics
m	Number of users at level 1
n	Number of items at level 1
l	Number of levels
$R^{(\ell)}$	Rating matrix at level ℓ
$S^{(\ell)}$	Ratings of user groups at level $\ell+1$ to item groups at level ℓ
$V^{(\ell)}$	Ratings of user groups at level ℓ to item groups at level $\ell+1$
Q	d-dimensional latent factors for user groups ($m \times d$)
P	d-dimensional latent factors for item groups ($n \times d$)
θ	d-dimensional topic distribution for user groups ($m \times d$)
μ	d-dimensional topic distribution for item groups ($n \times d$)
ϕ	User group distribution per topic ($d \times m$)
κ	Item group distribution per topic ($d \times n$)
g_{ui}	Topic assignment of user group u on item group i
h_{iu}	Topic assignment of item group i for user group u
e_u	Index of user group u's parent group in higher layer
t_i	Index of item group i's parent group in higher layer
$WQ_u^{(\ell)}$, $WP_i^{(\ell)}$	Child group of u/i at level ℓ
$FQ_u^{(\ell)}$, $FP_i^{(\ell)}$	Parent group of u/i at level ℓ

3.1 Basic Idea

Unlike previous hierarchical MF approaches, where only the rating error is minimized at each level, HHMF tries to maximize both the probability that the training corpus R is generated and the probability for the grouping of users and items. HHMF includes two steps: (i) learning the hidden structure from the bottom layer to the top layer of the hierarchy and then (ii) predicting ratings from the top layer to the bottom layer, as illustrated in Figure 2. Later, we will explain the two steps of this process in details.

There are two types of latent variables in HHMF:

– The *latent factors* Q_u and P_i of a user group u and an item group i in HHMF, respectively. These are similar to the latent factors in traditional MF, modeling the latent preferences of a single user and the latent properties of a single item. Each dimension in a Q_u and a P_i can take any value.

– The *latent topics* θ_u and μ_i of a user group u and an item group i in HHMF, respectively. These are the latent topic distributions of the user and the item groups and can also be regarded as their latent affiliation to larger groups with latent topics (e.g., horror movies and comedy movies). The value of each dimension in θ_u or μ_i represents the probability that user group u or item group i belongs to the corresponding larger group with a certain latent topic. Therefore, the sum of values of all dimensions in θ_u or μ_i should be equal to 1.

The idea behind learning the hidden structure is that the latent factor vectors Q_u and P_i are likely to 'match' the latent topic distribution vectors θ_u

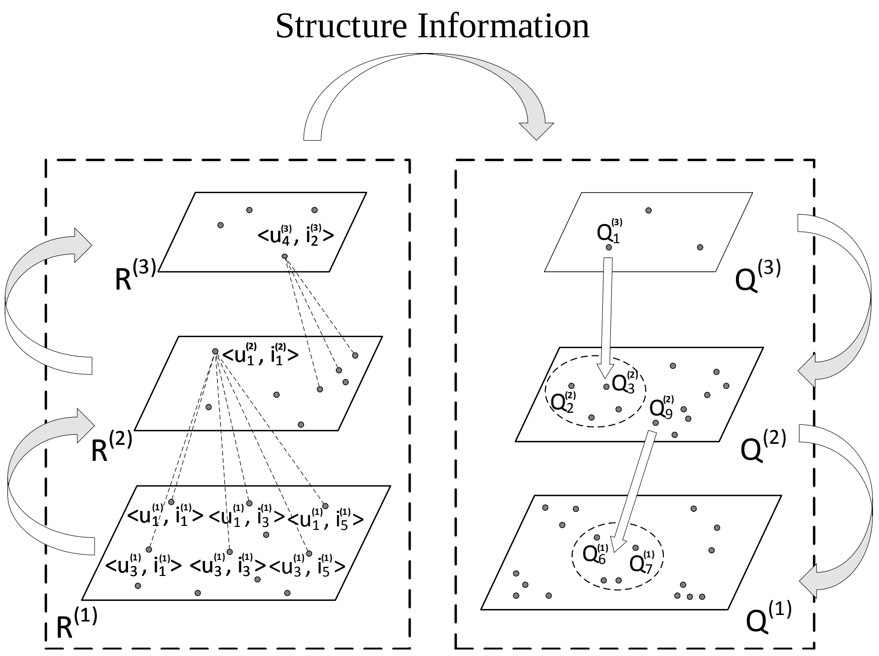


Fig. 2 Overview of a three-level HHMF. The left part indicates the structure learning phase, while the right part demonstrates the prediction phase. $u_4^{(3)}$ indicates user group 4 in layer 3 and $Q_1^{(3)}$ denotes user latent vector of user group 1 in layer 3. $R^{(1)}$ is the original $m \times n$ rating matrix. To plot Q in the right $2d$-dimensional space, we assume $d = 2$ and each Q becomes a point in the space. P is similarly learned and therefore omitted. WQ/FQ are child/parent groups of a user group and e is the index of the parent group of a user group. For example, $WQ_1^{(2)} = \{u_1^{(1)}, u_3^{(1)}\}$, $FQ_1^{(1)} = u_1^{(2)}$ and $e_1^{(1)} = 1$. WP/FP and t are defined similarly.

and μ_i of the corresponding user group u and item group i. Specifically, if user group u prefers comedy movies more, the probability of this topic in θ_u will be high. By linking factors and topics together, we hope that if a user group 'likes' a certain property (high value in certain dimensions in Q_u), this will correspond to a high probability for corresponding topics (high value in certain dimensions in θ_u). Similarly, if an item group exhibits a certain property, this will correspond to some particular topics being observed in μ_i. After θ and μ are learned, it is easy to obtain the hierarchical structure of user groups and item groups (e.g., a fine-grained to coarse-grained hierarchy, such as *tracks*, *albums*, *artists* and *genres*) using various methods (e.g., by choosing the topic with the highest probability in θ or μ).

After the hierarchy has been discovered, HHMF combines all the correlations introduced in Section 2.2.1, and the latent factors are generated from higher layers to lower layers. *Correlation representation* (Shan et al. 2012; Wang et al. 2014) which constructs new rating matrices for each layer is adopted in order to supervise the learning of Q and P at each layer. During this step, Q and P already incorporate the grouping information from latent topics through hierarchical grouping.

HHMF relies on both latent factors and latent topics. Latent factors have the same functionality as standard MF, while a 'latent topic' has two roles in HHMF: (1) it helps the model to learn the hidden hierarchical structure; (2) it helps to reduce the prediction error of MF.

In the following, we will first describe in detail the two processes in HHMF. Then, the overall hierarchical and iterative procedure of HHMF will be presented.

3.2 Upstream Hidden Structure Learning

In its first phase (structure learning), HHMF tries to discover the latent hierarchical affiliation of user groups and item groups. The objective of this learning phase is to maximize the probability for the grouping of users and items (via latent topics) and minimize the prediction error at each layer (via latent vectors) in the discovered hierarchy.

For each user group u and item group i, we learn a topic distribution θ_u and μ_i (i.e., a stochastic vector in Δ^d). These vectors encode the extent to which each of the d topics is preferred across all rating records for user group u and item group i, respectively. Each topic z is also associated with a user group distribution ϕ_z and an item group distribution κ_z, which encodes the probability that a particular user group (resp. item group) shows preference for (resp. relation to) topic z.

Based on our motivation, we do not design HHMF to learn latent factors and latent topics independently. Instead, by *linking* the two types of latent variables, if a user group u exhibits certain preference in latent factors Q_u, we anticipate that the corresponding topic is observed in the latent topics θ_u and certain user group distributions ϕ. Similarly, latent factors P_i and latent topics μ_i (and certain item distributions κ) are linked together. In other words, HHMF is different than traditional supervised topic models (Blei et al. 2003) which learns topics correlated with an output variable. In HHMF, latent topics are learned in a way such that they are correlated with latent factors.

Now the problem is how we link the two latent variables. A straightforward approach is to enforce each dimension in θ_u (or μ_i) to be equal to the corresponding dimension in Q_u (or P_i). However, such a method will definitely make the model to lose its expressive power. As discussed in Section 3.1, latent factors and latent topics have different value ranges for each dimension. Each of the entries in θ_u or μ_i describes the probability that the user group or item group is related to a certain topic, and thus the value should be less than or equal to 1. On the other hand, there are no constraints of the values in latent factors. In addition to the expressiveness, the desired link should be *monotonic*, i.e., the link should preserve the order so that large values in Q_u (or P_i) correspond to large values in θ_u (or μ_i). Considering both the expressiveness and monotonicity, our HHMF adopts the following link:

$$\theta_{(u,k)} = \frac{\exp(\epsilon \cdot Q_{uk})}{\sum_{b=1}^{d} \exp(\epsilon \cdot Q_{ub})}, \tag{2}$$

where ϵ is a parameter which controls the link and will be fitted during learning. As ϵ increases, θ_u converges to a unit vector which only takes the value 1 at the dimension corresponding to the most important topic. When ϵ decreases, user group u becomes more evenly affiliated to different topic groups. μ and P have a similar link and the parameter that controls the link is γ. The above links respect both the expressiveness (i.e., the range of a probability) and the monotonicity. In addition to linking θ (or μ) to Q (or P), we have to connect Q (or P) to ϕ (or κ) as well. Note that θ and ϕ are different, even though ϕ is obtained in a similar way as Equation 2. Specifically, we have $\sum_z \theta_{uz} = 1$ for vector θ_u and $\sum_u \phi_{zu} = 1$ for vector ϕ_z. The same holds for μ and κ. It is notable that we implicitly assume the number of factors of user/item groups is the same as the number of topics.

Given the links between latent factors and latent topics, HHMF maximizes the following posterior probability:

$$Pr(Q, P, \theta, \mu, \phi, \kappa | R, \Xi) \propto Pr(Q, P | R, \Xi) \cdot Pr(R|\theta, \kappa) \cdot Pr(R|\mu, \phi), \tag{3}$$

where $\Xi = \{\sigma_Q^2, \sigma_P^2\}$ are the standard deviations and R is the rating matrix. For the sake of readability, we put the calculation of Equation 3 in Appendix 6.1. When deducing Equation 3, we assume θ, ϕ and Q are conditionally independent given R (see the third step in Appendix 6.1) and this makes the problem simpler. A similar assumption holds for μ, κ and P.

We adopt the method used in probabilistic matrix factorization (PMF) (Salakhutdinov and Mnih 2007) to estimate the first term $Pr(Q, P|R, \Xi)$ in Equation 3. In PMF, zero-mean spherical Gaussian priors are placed over user and item latent vectors:

$$Pr(Q|\sigma_Q^2) = \prod_{u=1}^{m} \mathcal{N}(Q_u|\sigma_Q^2 \mathrm{I}), \quad Pr(P|\sigma_P^2) = \prod_{i=1}^{n} \mathcal{N}(P_i|\sigma_P^2 \mathrm{I}). \tag{4}$$

To estimate the second and third terms $Pr(R|\theta, \kappa)$ and $Pr(R|\mu, \phi)$, we firstly define R in another format δ: δ is the set of $\langle$user group, item group$\rangle$ records, i.e., $\delta_{(u,i)} = 1$ if user group u has rated item group i and $\delta_{(u,i)} = 0$ otherwise. Then, $Pr(R|\theta, \kappa) = Pr(\delta|\theta, \kappa)$ and $Pr(R|\mu, \phi) = Pr(\delta|\mu, \phi)$. Parameters $\theta, \mu, \phi, \kappa$ can be updated via sampling (Blei et al. 2003). The likelihood of generating record set δ is:

$$\begin{aligned} Pr(\delta|\theta, \kappa) &= \mathit{\Pi}_{\langle u,i\rangle \in \delta}\ \theta_{(u,g_{ui})} \cdot \kappa_{(g_{ui},i)} \\ Pr(\delta|\mu, \phi) &= \mathit{\Pi}_{\langle u,i\rangle \in \delta}\ \mu_{(i,h_{iu})} \cdot \phi_{(h_{iu},u)} \end{aligned} \tag{5}$$

where g_{ui} is the topic assignment of user group u on item group i and h_{iu} is the topic assignment of item group i for user group u. Then, the log of the posterior distribution in Equation 3 can be written as:

$$\begin{aligned}
\mathcal{L} &= \ln\Big(Pr(Q,P,\theta,\mu,\phi,\kappa|R,\varXi)\Big) \\
&\propto \ln\Big(Pr(Q,P|R,\varXi)\Big) + \ln\Big(Pr(\delta|\theta,\kappa)\Big) + \ln\Big(Pr(\delta|\mu,\phi)\Big) \\
&= -\frac{1}{2\sigma_R^2}\sum_{r_{ui}\in R}(r_{ui}-\langle Q_u,P_i\rangle)^2 - \frac{1}{2\sigma_Q^2}\sum_{u=1}^{m}||Q_u||^2 - \frac{1}{2\sigma_P^2}\sum_{i=1}^{n}||P_i||^2 \\
&\quad -\frac{1}{2}\big(|R|\ln\sigma_R^2 + dm\ln\sigma_Q^2 + dn\ln\sigma_P^2\big) + Const \\
&\quad + \sum_{u=1}^{m}\sum_{z=1}^{d}C_u^z\ln(\theta_{(u,z)}) + \sum_{z=1}^{d}\sum_{i=1}^{n}C_z^i\ln(\kappa_{(z,i)}) \\
&\quad + \sum_{i=1}^{n}\sum_{z=1}^{d}C_i^z\ln(\mu_{(i,z)}) + \sum_{z=1}^{d}\sum_{u=1}^{m}C_z^u\ln(\phi_{(z,u)}),
\end{aligned} \tag{6}$$

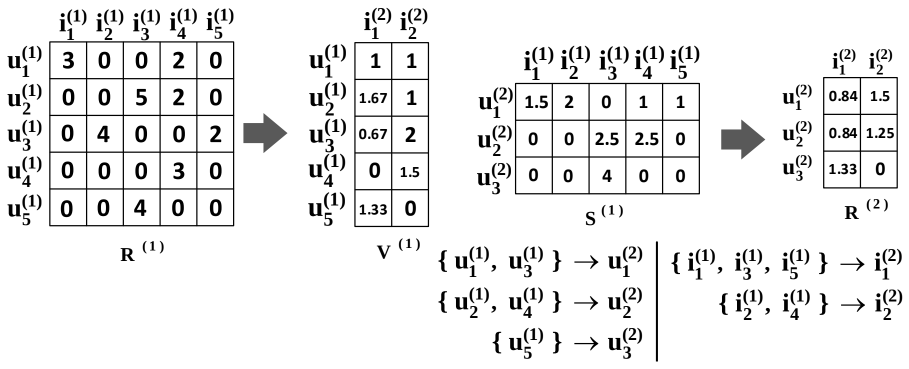
where $Const$ is a constant that does not depend on the parameters, and $|R|$ is the number of ratings in R. We define C_u^z as the number of times that topic z has been observed with a rating for the user group u, and C_z^i as the number of times the item group i is assigned to topic z. Similarly, we have C_i^z and C_z^u.

The first component $\ln\Big(Pr(Q,P|R,\varXi)\Big)$ in Equation 6 is traditionally optimized by gradient descend based methods or alternating least-squares (Koren et al. 2009), while iterative sampling is used to solve the second and third terms $\ln\Big(Pr(\delta|\theta,\kappa)\Big)$ and $\ln\Big(Pr(\delta|\mu,\phi)\Big)$ (Blei et al. 2003). Algorithm 1 demonstrates how HHMF discovers the latent hierarchy. To optimize the log of the posterior distribution in Equation 6 which consists of these components together, we first fix the topic assignments g/h and fit Q, P, θ, μ, ϕ, κ, ϵ and γ through L-BFGS, a quasi-Newton method for non-linear optimization of problems with many variables (Nocedal 1980). Then, we iterate through all training ratings and update their topic assignments using the sampling method: for each training rating r_{ui}, HHMF samples the topic assignment g once with probability $Pr(g_{ui}=b)=\theta_{(u,b)}\cdot\kappa_{(b,i)}$ and then it samples h with probability $Pr(h_{iu}=b)=\mu_{(i,b)}\cdot\phi_{(b,u)}$. Note that Q and θ are linked through Equation 2, thus they are optimized interdependently. In HHMF, we enforce that a change in Q affects both latent factors and latent topics during L-BFGS optimization. Similarly, a change in P will have an impact on both latent variables.

From the lowest to the highest layer of the hierarchy, HHMF iteratively learns *latent factors* and *latent topics* for each layer. The corresponding group assignment of a specific user group/item group is then the topic with the maximum probability in its *latent topic vector*, i.e., $e_u \leftarrow \arg\max_z(\theta_{(u,z)})$ and $t_i \leftarrow \arg\max_z(\mu_{(i,z)})$. Then, the ratings of a group at any layer are computed by averaging the ratings of its members from the layer below, which is often called *correlation representation* (see Figure 3) in hierarchical MF (Shan et al. 2012; Wang et al. 2014). Particularly, $R^{(1)}$ is the original rating matrix, which represents $(u^{(1)}, i^{(1)})$ correlation. The rating matrices generated by $(u^{(1)}, i^{(2)})$

Algorithm 1 Bottom-up Hidden Structure Learning

```
1:  for $\ell$ in $1 \ldots l$ do
2:      while Not Converge do
3:          Fix $g$ and $h$, then optimize Equation 6 to learn $Q, P, \theta, \mu, \phi, \kappa$ using L-BFGS.
4:          for $r_{ui}$ in $R^{(\ell)}$ do
5:              Samples topic assignment $g_{ui}$ with probability $Pr(g_{ui} = b) = \theta_{(u,b)} \cdot \kappa_{(b,i)}$.
6:              Samples topic assignment $h_{iu}$ with probability $Pr(h_{iu} = b) = \mu_{(i,b)} \cdot \phi_{(b,u)}$.
7:          end for
8:          for $u$ in $U$ do
9:              $e_u \leftarrow \arg\max_z(\theta_{(u,z)})$
10:         end for
11:         for $i$ in $I$ do
12:             $t_i \leftarrow \arg\max_z(\mu_{(i,z)})$
13:         end for
14:     end while
15:     Form $R^{(\ell+1)}$ using grouping and $R^{(\ell)}$.
16: end for
```



$\begin{array}{c|ccccc} & i_1^{(1)} & i_2^{(1)} & i_3^{(1)} & i_4^{(1)} & i_5^{(1)} \\ \hline u_1^{(1)} & 3 & 0 & 0 & 2 & 0 \\ u_2^{(1)} & 0 & 0 & 5 & 2 & 0 \\ u_3^{(1)} & 0 & 4 & 0 & 0 & 2 \\ u_4^{(1)} & 0 & 0 & 0 & 3 & 0 \\ u_5^{(1)} & 0 & 0 & 4 & 0 & 0 \end{array}$
R (1)
$\begin{array}{c|cc} & i_1^{(2)} & i_2^{(2)} \\ \hline u_1^{(1)} & 1 & 1 \\ u_2^{(1)} & 1.67 & 1 \\ u_3^{(1)} & 0.67 & 2 \\ u_4^{(1)} & 0 & 1.5 \\ u_5^{(1)} & 1.33 & 0 \end{array}$
V (1)
$\begin{array}{c|ccccc} & i_1^{(1)} & i_2^{(1)} & i_3^{(1)} & i_4^{(1)} & i_5^{(1)} \\ \hline u_1^{(2)} & 1.5 & 2 & 0 & 1 & 1 \\ u_2^{(2)} & 0 & 0 & 2.5 & 2.5 & 0 \\ u_3^{(2)} & 0 & 0 & 4 & 0 & 0 \end{array}$
S (1)
$\begin{array}{c|cc} & i_1^{(2)} & i_2^{(2)} \\ \hline u_1^{(2)} & 0.84 & 1.5 \\ u_2^{(2)} & 0.84 & 1.25 \\ u_3^{(2)} & 1.33 & 0 \end{array}$
R (2)
$\{u_1^{(1)}, u_3^{(1)}\} \rightarrow u_1^{(2)}$
$\{u_2^{(1)}, u_4^{(1)}\} \rightarrow u_2^{(2)}$
$\{u_5^{(1)}\} \rightarrow u_3^{(2)}$
$\{i_1^{(1)}, i_3^{(1)}, i_5^{(1)}\} \rightarrow i_1^{(2)}$
$\{i_2^{(1)}, i_4^{(1)}\} \rightarrow i_2^{(2)}$

Fig. 3 A toy example of Correlation Representation in Hierarchical MF (Shan et al. 2012; Wang et al. 2014). The figure shows the representations from level 1 to level 2 in the hierarchy of Figure 2.

and $(i^{(1)}, u^{(2)})$ are denoted as $V^{(1)}$ and $S^{(1)}$, respectively. Given $V^{(1)}$ and $S^{(1)}$, the correlation $(u^{(2)}, i^{(2)})$ can be obtained and it is defined as $R^{(2)}$, i.e., a higher level rating matrix compared to $R^{(1)}$. The definition can be generalized to $R^{(l)}$, $S^{(l)}$ and $V^{(l)}$ where l is the layer index.

To better understand the correlation representation, we now go through the hierarchy in Figure 3 from its highest to its lowest level (i.e., $R^{(2)} \rightarrow \{V^{(1)}, S^{(1)}\} \rightarrow R^{(1)}$). To illustrate, we pick the record $r_{11}^{(2)} = \langle u_1^{(2)}, i_1^{(2)} \rangle = 0.84$ at the second layer (i.e., $r_{11}^{(2)} \in R^{(2)}$) as the starting point. $r_{11}^{(2)}$ represents the average aggregation of $s_{11}^{(1)} = \langle u_1^{(2)}, i_1^{(1)} \rangle = 1.5$, $s_{13}^{(1)} = \langle u_1^{(2)}, i_3^{(1)} \rangle = 0$ and $s_{15}^{(1)} = \langle u_1^{(2)}, i_5^{(1)} \rangle = 1$ (i.e., $(1.5 + 0 + 1)/3 = 0.84$) in $S^{(1)}$, since $i_1^{(2)}$ contains $\{i_1^{(1)}, i_3^{(1)}, i_5^{(1)}\}$. $r_{11}^{(2)}$ can also be interpreted as the average aggregation of $v_{11}^{(1)} = \langle u_1^{(1)}, i_1^{(2)} \rangle = 1$, $v_{31}^{(1)} = \langle u_3^{(1)}, i_1^{(2)} \rangle = 0.67$ in $V^{(1)}$ (i.e., $(1 + 0.67)/2 = 0.84$), since $u_1^{(2)}$ contains $\{u_1^{(1)}, u_3^{(1)}\}$. $S^{(1)}$ and $V^{(1)}$ can also be computed using the aforementioned definition, e.g., $s_{14}^{(1)} = (r_{14}^{(1)} + r_{34}^{(1)})/2 = (2+0)/2 = 1$ and $v_{31}^{(1)} = (r_{31}^{(1)} + r_{33}^{(1)} + r_{35}^{(1)})/3 = (0+0+2)/3 = 0.67$. Note that the correlation representation can be interpreted from an inverse direction, i.e., from the lowest to the highest level (i.e., $R^{(1)} \to \{V^{(1)}, S^{(1)}\} \to R^{(2)}$), and the same scores (representations) can be obtained.

HHMF discovers the affiliation for user groups and item groups, which minimizes the prediction error based on the group ratings at each layer in the hierarchy. This bottom-up procedure forms the fine-grained to coarse-grained hierarchy.

3.3 Downstream Rating Prediction

HHMF applies a top-down approach to perform recommendations, using the hierarchy learned in the first phase. In this phase, HHMF becomes a generative model.

Specifically, the user-group latent vectors $\{Q^{(\ell)}\}_{\ell=1}^{l}$ and the item-group latent vectors $\{P^{(\ell)}\}_{\ell=1}^{l}$ are used to predict the unobserved ratings at different levels. These are generated using the links and the ratings from the higher level. Similarly to HGMF, HHMF assumes that the rating matrices $\{R^{(\ell)}\}_{\ell=1}^{l}$, $\{S^{(\ell)}\}_{\ell=1}^{l-1}$ and $\{V^{(\ell)}\}_{\ell=1}^{l-1}$ are drawn from a higher layer to a lower layer. Differently, the probability of generating $Q_u^{(\ell)}$ (i.e., $Pr(Q_u^{(\ell)}|Q^{(\ell+1)})$) consists of two parts, $\mathcal{N}(Q^{(\ell+1)}, \sigma^2_{Q^{(\ell+1)}}\mathrm{I})$ and $\phi^{(\ell+1)}$. Here, $\phi^{(\ell+1)}$ is the user group distribution per topic (d-dimensional vector) and it can be regarded as the confidence that the specific user group u is generated by the corresponding user group e_u. Similarly, we can get $Pr(P_i^{(\ell)}|P^{(\ell+1)})$.

The generative process[8] can be illustrated as follows:

- For each layer $\ell = l, \dots, 1$
  1. For each user group u in layer ℓ, generate $Q_u^{(\ell)} \sim \mathcal{N}(Q_{e_u}^{(\ell+1)}, \sigma_{Q^{(\ell+1)}}{}^2\mathrm{I}) \cdot \phi^{(\ell+1)}_{(e_u,u)}$
  2. For each item group i in layer ℓ, generate $P_i^{(\ell)} \sim \mathcal{N}(P_{t_i}^{(\ell+1)}, \sigma_{P^{(\ell+1)}}{}^2\mathrm{I}) \cdot \kappa^{(\ell+1)}_{(t_i,i)}$
- For each layer $\ell = l, \dots, 1$
  1. Draw $r_{ui}^{(\ell)} \sim \mathcal{N}(\langle Q_u^{(\ell)}, P_i^{(\ell)}\rangle, \sigma^2_{R^{(\ell)}})$ for every observed rating r_{ui} of $R^{(\ell)}$.
  2. Draw $s_{ui}^{(\ell)} \sim \mathcal{N}(\langle Q_u^{(\ell+1)}, P_i^{(\ell)}\rangle, \sigma^2_{S^{(\ell)}})$ for every observed rating s_{ui} of $S^{(\ell)}$.
  3. Draw $v_{ui}^{(\ell)} \sim \mathcal{N}(\langle Q_u^{(\ell)}, P_i^{(\ell+1)}\rangle, \sigma^2_{V^{(\ell)}})$ for every observed rating v_{ui} of $V^{(\ell)}$.

[8] $Q^{(l+1)} = P^{(l+1)} = 0$.

For example, the arrow from $Q_1^{(3)}$ to the dashed circle containing $Q_2^{(2)}$ and $Q_3^{(2)}$ in Figure 2 indicates that $u_2^{(2)}$ and $u_3^{(2)}$ belong to the same group $u_1^{(3)}$ in the layer above and their latent vectors are generated from $Q_1^{(3)}$ at the layer above.

The posterior probability of $\{Q^{(\ell)}\}_{\ell=1}^{l}$ and $\{P^{(\ell)}\}_{\ell=1}^{l}$ is:

$$\begin{aligned}
&Pr(\{Q^{(\ell)}\}_{\ell=1}^{l}, \{P^{(\ell)}\}_{\ell=1}^{l} \mid \{R^{(\ell)}\}_{\ell=1}^{l}, \{S^{(\ell)}\}_{\ell}^{l-1}, \{V^{(\ell)}\}_{\ell=1}^{l-1}, \varXi)\\
&\propto \prod_{\ell=1}^{l} \prod_{r_{ui}\in R^{(\ell)}} \mathcal{N}(\langle Q_u^{(\ell)}, P_i^{(\ell)}\rangle, \sigma^2_{R^{(\ell)}}) \cdot \prod_{\ell=1}^{l-1} \prod_{s_{ui}\in S^{(\ell)}} \mathcal{N}(\langle Q_u^{(\ell+1)}, P_i^{(\ell)}\rangle, \sigma^2_{S^{(\ell)}})\\
&\quad \prod_{\ell=1}^{l-1} \prod_{v_{ui}\in V^{(\ell)}} \mathcal{N}(\langle Q_u^{(\ell)}, P_i^{(\ell+1)}\rangle, \sigma^2_{V^{(\ell)}}) \cdot \prod_{\ell=1}^{l} \prod_{u=1}^{m^{(\ell)}} \phi^{(\ell+1)}_{(e_u,u)} \mathcal{N}(Q^{(\ell+1)}, \sigma^2_{Q^{(\ell+1)}}\mathrm{I})\\
&\quad \prod_{\ell=1}^{l} \prod_{i=1}^{n^{(\ell)}} \kappa^{(\ell+1)}_{(t_i,i)} \mathcal{N}(P^{(\ell+1)}, \sigma^2_{P^{(\ell+1)}}\mathrm{I}),
\end{aligned} \tag{7}$$

where $\varXi = \{\sigma_Q^2, \sigma_P^2, \sigma_R^2, \sigma_S^2, \sigma_V^2\}$. To improve the readability, we put the calculation of Equation 7 in Appendix 6.2.

The log of the posterior distribution over the user-group and item-group latent features is given by:

$$\begin{aligned}
\mathcal{L} &= \ln\Big(P(\{Q^{(\ell)}\}_{\ell=1}^{l}, \{P^{(\ell)}\}_{\ell=1}^{l} \mid \{R^{(\ell)}\}_{\ell=1}^{l}, \{S^{(\ell)}\}_{\ell=1}^{l-1}, \{V^{(\ell)}\}_{\ell=1}^{l-1}, \varXi)\Big)\\
&\propto -\sum_{\ell=1}^{l} \frac{1}{2\sigma^2_{R^{(\ell)}}} \sum_{r_{ui}\in R^{(\ell)}} (r_{ui} - \langle Q_u^{(\ell)}, P_i^{(\ell)}\rangle)^2 - \sum_{\ell=1}^{l-1} \frac{1}{2\sigma^2_{S^{(\ell)}}} \sum_{s_{ui}\in Q^{(\ell)}} (s_{ui} - \langle Q_u^{(\ell+1)}, P_i^{(\ell)}\rangle)^2\\
&\quad -\sum_{\ell=1}^{l-1} \frac{1}{2\sigma^2_{V^{(\ell)}}} \sum_{v_{ui}\in V^{(\ell)}} (v_{ui} - \langle Q_u^{(\ell)}, P_i^{(\ell+1)}\rangle)^2 - \sum_{\ell=1}^{l} \frac{1}{2\sigma^2_{Q^{(\ell+1)}}} \sum_{u=1}^{m^{(\ell)}} \phi^{(\ell+1)}_{(e_u,u)} (Q_u^{\ell} - Q_{e_u}^{(\ell+1)})^2\\
&\quad -\sum_{\ell=1}^{l} \frac{1}{2\sigma^2_{P^{(\ell+1)}}} \sum_{i=1}^{n^{(\ell)}} \kappa^{(\ell+1)}_{(t_i,i)} (P_i^{\ell} - P_{t_i}^{(\ell+1)})^2.
\end{aligned} \tag{8}$$

As in HGMF, we can perform a stochastic block co-ordinate ascent algorithm (Bertsekas 2006) to update $\{Q^{(\ell)}\}_{\ell=1}^{l}$ and $\{P^{(\ell)}\}_{\ell=1}^{l}$, i.e., Q and P will be updated sequentially at each layer for fast convergence. Algorithm 2 illustrates the update procedure. The update rules can be obtained using the partial derivative of Equation 8:

$$\varTheta^{(\ell)} \leftarrow \varTheta^{(\ell)} + \eta \cdot \mathit{Gradient}, \tag{9}$$

where η is the learning rate and $\varTheta^{(\ell)}$ indicates the parameter to be updated (i.e., Q and P). *Gradients* are the partial derivatives of Equation 8, which we put in Appendix 6.3 for better readability.

Algorithm 2 Top-down Rating Prediction

```
1: for $\ell$ in $1, \dots, l-1$ do
2:     $S^{(\ell)} \leftarrow S^{(\ell)} - \bar{S}^{(\ell)}$
3:     $V^{(\ell)} \leftarrow V^{(\ell)} - \bar{V}^{(\ell)}$
4:     $R^{(\ell)} \leftarrow R^{(\ell)} - \bar{R}^{(\ell)}$
5: end for
6: $R^{(\ell)} \leftarrow R^{(\ell)} - \bar{R}^{(\ell)}$
7: while Not Converge do
8:     for $\ell$ in $l \dots 2$ do
9:         Sequentially update $Q^{(\ell)}$, $P^{(\ell)}$, $Q^{(\ell)}$, $P^{(\ell-1)}$, $Q^{(\ell-1)}$, $P^{(\ell)}$ using gradients in Equations 16, 17, 18, 19, 20 and 21, respectively.
10:    end for
11:    Sequentially update $Q^{(1)}$, $P^{(1)}$ using gradients in Equations 16 and 17, respectively.
12: end while
13: Output the prediction matrix as $\langle Q^{(1)}, P^{(1)} \rangle + \bar{R}^{(1)}$.
```

Algorithm 3 HHMF

```
1: // Bottom-up Hidden Structure Learning
2: for $\ell$ in $1 \dots l$ do
3:     while Not Converge do
4:         Fix $g$ and $h$, optimize Equation 6 to learn $Q, P, \theta, \mu, \phi, \kappa$ using L-BFGS.
5:         Sample $g$ and $h$ to update topic assignment.
6:     end while
7:     Form the groups using $\theta$ and $\mu$.
8: end for
9:
10: // Top-down Rating Prediction
11: for $\ell$ in $l \dots 1$ do
12:    Optimize Equation 8 using batch gradient ascent algorithm as illustrated in Algorithm 2.
13: end for
14: Output the prediction matrix as $\langle Q^{(1)}, P^{(1)} \rangle + \bar{R}^{(1)}$.
```

3.4 HHMF: Putting It All Together

The overall HHMF process is shown in Algorithm 3. HHMF first learns the hidden hierarchical structure bottom-up (lines 2-8). Based on the obtained structural information, HHMF predicts the ratings in a top-down fashion (lines 11-14).

Time complexity. Hierarchical MF methods are inherently slower than single-layer MF due to the traversal of multiple layers (e.g., lines 2 and 11 in Algorithm 3). HGMF (Wang et al. 2014) has a time complexity of $O\big(l(\bar{m}^2+\bar{n}^2)\big)$ for its clustering phase due to the calculation of the similarity between all pairs of users and $O\big(dl(\bar{m}+\bar{n})\big)$ for the prediction phase, where $\bar{m}$ and $\bar{n}$ represent the average number of users and items at each level, respectively. The time complexity of the bottom-up structure learning phase in HHMF is $O\big(\frac{l(\bar{m}^2+\bar{n}^2+\bar{m}+\bar{n})}{2}\big)$, if L-BFGS (Nocedal 1980) is used for optimization. Our HHMF method has the same time complexity in its top-down rating prediction phase as the learning phase of HGMF, if both methods use the same gradient ascent approach. Overall, HHMF has the same time complexity as HGMF, while it significantly outperforms HGMF in terms of recommendation quality, as we will show in Section 4.

4 Empirical Study

In this section, we conduct an experimental study using real datasets to compare the performance of HHMF with the state-of-the-art hierarchical MF methods and other traditional methods for recommender systems in order to answer the following four questions:

- **Q1**: Does the proposed HHMF model outperform the state-of-the-art hierarchical MF approaches?
- **Q2**: Is HHMF sensitive to its hyperparameters? In other words, can HHMF be easily tuned so that it keeps outperforming other models?
- **Q3**: Is HHMF an efficient method which can be used in practice?
- **Q4**: Is the learned ‘hidden’ hierarchy reasonable in practice?

4.1 Dataset

Six public datasets are used in our experiments: FilmTrust, Ciao, Yelp, Dianping, MovieLens and Netflix:

- **FilmTrust**[9] is a website that integrates semantic web-based social networks, augmented with trust, to create predictive movie recommendations. The dataset is provided by Guo et al. (2013) and it was collected in June 2011.
- **Ciao**[9] is a platform which combines consumer reviews and up-to-date price information from hundreds of online merchants to provide a comprehensive source of shopping intelligence on the web. The dataset was crawled from the entire category of DVDs from Ciao UK website in December, 2013 (Guo et al. 2014).
- **Yelp**[10] is a local business recommender system and service provider. The dataset is from the Yelp Business Rating Prediction Challenge, which includes customers’ ratings on restaurants in Phoenix, United States.
- **Dianping**[11] is a social network-based recommender system, which is a leading local restaurant search and review platform in China. The dataset contains business items, user information in Shanghai and the ratings from April 2003 to November 2013. We use the version provided by Li et al. (2014).

[9] http://www.librec.net/datasets.html
[10] http://www.kaggle.com/c/yelp-recsys-2013
[11] http://lihui.info/data/dianping.html

Table 2 Statistics of Data

Dataset	# of users at level 1	# of items at level 1	# of ratings at level 1
FilmTrust	1,508	2,071	35,497
Ciao	17,615	16,121	72,664
Yelp	43,873	11,537	252,863
Dianping	11,352	10,657	501,472
MovieLens	71,567	10,681	10,000,054
Netflix	480,189	17,770	100,480,507

– **MovieLens**[12] is an online movie recommender system and virtual community that recommends movies to users. The recommendation is based on their film preferences, using collaborative filtering based on members' movie ratings. We use MovieLens 10M dataset, which is one of the standard benchmark datasets of MovieLens (Harper and Konstan 2016).
– **Netflix**[13] is a media services provider which offers film and shows to its subscribers. We use the dataset which was provided by Netflix Prize. Netflix Prize was an open competition for collaborative filtering algorithms that predict user ratings for films (Bell and Koren 2007).

Table 2 shows some general statistics of the datasets. For datasets FilmTrust, Ciao and Dianping, we randomly selected 80% ratings in each dataset to be used for training; the remaining 20% ratings are held out for testing. For datasets Yelp, MovieLens and Netflix, we used the official training/test split.

4.2 Performance Metrics

We adopt several measures including Root Mean Square Error (RMSE), Precision@k, Recall@k, F1@k and NDCG@k to measure the quality of recommendation, since they are widely used in the evaluation of recommender systems (Herlocker et al. 2004). RMSE measures the accuracy of predicted ratings. However, modeling only observed ratings is insufficient for a ranking based (i.e., top-k) recommender system which is more common nowadays (Hu et al. 2008). Therefore, we additionally use Precision@k, Recall@k, F1@k and NDCG@k to evaluate the performance regarding the top-k recommendation. The aforementioned measures are defined as follows:

$$RMSE = \sqrt{\frac{1}{|R|}\sum_{r_{ui}\in R}(r_{ui} - \hat{r}_{ui})^2}, \tag{10}$$

where $|R|$ denotes the number of tested ratings, r_{ui} is a real rating, and $\hat{r}_{ui}$ is a predicted rating.

[12] https://grouplens.org/datasets/movielens
[13] https://www.netflixprize.com

$$Precision@k = \frac{1}{m}\sum_{i=1}^{m}\frac{|G_{u_i}(k) \cap T_{u_i}(k)|}{|T_{u_i}(k)|}, \tag{11}$$

$$Recall@k = \frac{1}{m}\sum_{i=1}^{m}\frac{|G_{u_i}(k) \cap T_{u_i}(k)|}{|G_{u_i}(k)|}, \tag{12}$$

where m is the number of users, $T_{u_i}(k)$ denotes the set of top-k recommended items for user u_i, and $G_{u_i}(k)$ represents the true set of relevant items (ground-truth positives in test data). Given Precision@k and Recall@k, F1@k is the harmonic mean between them:

$$F_1@k = \frac{2 \times Precision@k \times Recall@k}{Precision@k + Recall@k}. \tag{13}$$

$$NDCG@k = \frac{1}{Z_k M}\sum_{i=1}^{k}\frac{2^{t_i}-1}{log_2(i+1)}, \tag{14}$$

where Z_k is a normalizer which ensures that perfect ranking has a value of 1; t_i is the relevance of item at position i. We use simple binary relevance: $t_i = 1$ if the item is in the test set, and 0 otherwise.

4.3 Competitors

In our evaluation, we compare the effectiveness of the following approaches:

- **KNN**: The k nearest neighbors algorithm is one of the most popular collaborative filtering methods, where the predicted ratings are given based on the ratings of the top-k most similar users via Cosine similarity (i.e., weighted sum).
- **PMF** (Salakhutdinov and Mnih 2007): This is the probabilistic matrix factorization method where only user-item correlation is considered.
- **FISM** (Kabbur et al. 2013)[14]: FISM learns the item-item similarity matrix as the product of two low dimensional latent factor matrices in order to overcome the problem of traditional nearest neighbor collaborative filtering methods, i.e., similarities between items which have not been co-rated by at least one user cannot be captured. FISM is motivated by NSVD (Paterek 2007) and SVD++ (Koren 2008) and it is a factored item similarity model.
- **IHSR** (Wang et al. 2015, 2018b)[15]: This is the weighted nonnegative MF (Zhang et al. 2006) based hierarchical MF approach. IHSR recursively performs nonnegative MF on the user preference matrix and item characteristic matrix and uses the nonnegative decomposed matrices to indicate the affiliations of users/items to different groups.

[14] `https://github.com/yushuai/FISM`
[15] `http://www.public.asu.edu/~swang187/codes/HSR.zip`

- **HGMF** (Wang et al. 2014): Hierarchical Group Matrix Factorization (HGMF) is the state-of-the-art hierarchical MF method, which outperforms a previously proposed HPMF method (Shan et al. 2012). We use the greedy-based clustering method suggested in (Wang et al. 2014) to obtain the structure information, required by HGMF as prior knowledge.
- **DMF** (Xue et al. 2017)[16]: Deep Matrix Factorization (DMF) combines MF and neural networks architecture. DMF uses a deep structure learning framework to learn the representations of users and items. The latent vectors are projected to another vector at each layer in order to learn the representation. Unlike other hierarchical MF approaches, the structure (i.e., layers in the neural network) in DMF does not represent the natural hierarchical structures of users and items.
- **NeuMF** (He et al. 2017)[17]: Neural Collaborative Filtering (NeuMF) is a neural network-based method which is similar to DMF. DMF adopts the user-item interaction matrix (i.e., rating matrix R) where each element corresponds to the real rating score, while NeuMF transforms R into a binary matrix, such that if a user has rated an item the corresponding element in R is 1.
- **NFM** (He and Chua 2017)[18]: Neural Factorization Machines combines the linearity of Factorization Machines (FMs) (Rendle 2010) in modeling second-order feature interactions and the non-linearity of neural networks in modeling higher-order feature interactions. We modified the original implementation in order for the method to be applicable for collaborative filtering.
- **HHMF**: HHMF is our proposed method which learns the hidden structure information and utilizes it to improve the quality of recommendation.

In order to demonstrate the effectiveness of HHMF, we evaluate it using only user-item rating information and ignore any contextual information such as user-generated text or check-in records, which may come together (and could be used to improve recommendation quality).

4.4 Hyperparameter Settings

A similar strategy for setting the hyperparameters of hierarchical MF as that in (Wang et al. 2014) is adopted: the learning rate η is set to 0.0005, the performances of HGMF and HHMF are tested for $\ell = 2, 3$. For the main experiments of HHMF, we set the maximum group size of each layer to be 10 times the size in the layer above and that of the highest layer to be equal to d, i.e., 10 groups for $\ell = 2$ and 100 groups for $\ell = 1$ if $d = 10$. We call this

[16] https://github.com/RuidongZ/Deep_Matrix_Factorization_Models
[17] https://github.com/hexiangnan/neural_collaborative_filtering
[18] https://github.com/hexiangnan/neural_factorization_machine

hyperparameter *group ratio* and we also report the impact of group ratio in Section 4.6.

For the factorization based methods (i.e., PMF, FISM, IHSR, HGMF, DMF, NeuMF, NFM and HHMF), values $\{0.0001, 0.001, 0.01, 0.1, 1, 10\}$ are tested for the regularization hyperparameter λ and we test values $\{10, 20, 30, 50\}$ for the hidden dimensionality d. For KNN the neighborhood size is tuned from values in $\{2, 3, 5, 10, 20\}$. 2 layers and a negative sampling ratio of 5 are used in DMF and NeuMF, following the authors' suggestions (Xue et al. 2017; He et al. 2017). The normalization constant in FISM is tuned from values in $\{0.4, 0.5, 0.6, 0.7, 0.8\}$, as suggested by Kabbur et al. (2013). We use 2 hidden layers for IHSR; the authors claimed that using more layers will result in similar observations (Wang et al. 2015, 2018b). For NFM, the best drop ratio is searched in the range $\{0, 0.1, 0.2, \cdots, 0.9\}$ as in the experiments conducted by He and Chua (2017).

5-fold cross-validation is performed on a small subset of the training data to empirically tune the hyperparameters so that each method achieves the best possible results. For all methods, except KNN, the experiment will terminate when the decrease of RMSE between two iterations becomes less than 0.001. The reported results are based on the best hyperparameters.

4.5 Quality of Recommendation (Q1)

We compare the effectiveness of the tested recommenders on two standard tasks: rating prediction and top-k recommendation, in order to answer question Q1.

4.5.1 Rating Prediction

The RMSE of all tested methods on rating prediction is shown in Tables 3 and 4. HHMF$_i$ implies that the results are obtained using HHMF with i levels. A similar notation is used for HGMF. HHMF$_i$ + HGMF$_i$ indicates that we run HHMF$_i$ once and use the learned hierarchy as input to HGMF$_i$. The best results of either HHMF$_2$ or HHMF$_3$ are indicated in bold. The last row indicates the improvement that HHMF achieves over the best competitor excluding HHMF$_i$ + HGMF$_i$. Note that HHMF outperforms the best runner-up by $4.15\% - 11.70\%$. Additionally, we can observe the following:

(1) MF based methods achieve better results than KNN in most cases.

(2) IHSR, HGMF and HHMF, which incorporate hierarchical structure information into MF, usually outperform PMF and FISM which only consider (user, item) and (item, item) correlations.

(3) HGMF and HHMF have better accuracy as the number of levels increases.

(4) HHMF significantly outperforms HGMF and IHSR when the same number of levels are used, showing its superiority over other hierarchical MF methods.

Table 3 RMSE on FilmTrust, Ciao and Yelp

Method	FilmTrust		Ciao		Yelp	
	d=10	d=20	d=10	d=20	d=10	d=20
KNN	1.001		1.1658		1.2656	
PMF	0.8992	0.9400	1.1895	1.1698	1.2120	1.2080
FISM	0.9211	0.9105	1.1512	1.1498	1.1876	1.1798
IHSR	0.9142	0.9098	1.1437	1.1412	1.1642	1.1579
NeuMF	0.8812	0.8743	1.0923	1.0781	1.1823	1.1809
DMF	0.8687	0.8513	1.1322	1.1218	1.1812	1.1798
NFM	0.8798	0.8762	1.1592	1.1582	1.1746	1.1708
$HGMF_2$	0.9067	0.8803	1.1773	1.1789	1.1982	1.1976
$HGMF_3$	0.9068	0.8997	1.1614	1.1422	1.1732	1.1687
$HHMF_2 + HGMF_2$	0.8335	0.8251	1.0854	1.0789	1.1654	1.1607
$HHMF_3 + HGMF_3$	0.8317	0.8219	1.0877	1.0642	1.1520	1.1354
$HHMF_2$	**0.8124**	0.8120	1.0417	1.0424	1.1398	1.1365
$HHMF_3$	0.8127	**0.8113**	**1.0255**	**1.0111**	**1.1023**	**1.0876**
Improvement	6.48%	4.70%	11.70%	6.21%	6.04%	6.94%

Table 4 RMSE on Dianping, MovieLens and Netflix

Method	Dianping		MovieLens		Netflix	
	d=10	d=20	d=10	d=20	d=10	d=20
KNN	0.8391		0.8742		0.9468	
PMF	0.7620	0.7667	0.8816	0.8812	0.9423	0.9388
FISM	0.7842	0.7541	0.8654	0.8578	0.9345	0.9311
IHSR	0.7762	0.7598	0.8164	0.8154	0.9311	0.9356
NeuMF	0.7654	0.7512	0.8244	0.8164	0.9314	0.9297
DMF	0.7732	0.7702	0.8106	0.8142	0.9267	0.9109
NFM	0.7684	0.7679	0.8348	0.8278	0.9266	0.9112
$HGMF_2$	0.7502	0.7748	0.8245	0.8194	0.9222	0.9235
$HGMF_3$	0.7498	0.7442	0.8187	0.8168	0.9211	0.9187
$HHMF_2 + HGMF_2$	0.7311	0.7568	0.8014	0.8120	0.9014	0.9123
$HHMF_3 + HGMF_3$	0.7298	0.7255	0.7987	0.7945	0.8925	0.8871
$HHMF_2$	0.7235	0.7242	0.7844	**0.7804**	0.8814	0.8810
$HHMF_3$	**0.7139**	**0.7041**	**0.7816**	0.7864	**0.8778**	**0.8645**
Improvement	4.79%	5.39%	4.26%	4.15%	4.70%	5.10%

(5) The learned ‘hidden’ hierarchy can be further used to improve the accuracy of HGMF, demonstrating that the two interdependent components in HHMF (i.e., rating prediction and hidden hierarchy discovery) mutually benefit each other and the learned structure has a better quality than the output structure from an independent structure discovery algorithm like the clustering algorithm used in HGMF.

(6) HHMF has significantly better performance than the state-of-the-art neural network-based methods NeuMF, DMF and NFM. The learned ‘hidden’ hierarchy in HHMF represents the similarity among users/items and therefore helps to improve the accuracy of prediction. As a comparison, a neural layer projects each latent factor vector to another vector in order to finally push the model towards a local optimum. Lacking a nat-

ural mapping between the projection and user/item correlations, neural network-based methods are inferior to HHMF.

In summary, the superior performance of HHMF over other approaches demonstrates the effectiveness of our proposal for the rating prediction task in recommender systems.

4.5.2 Top-k Recommendation

We choose three approaches (i.e., HGMF, NeuMF and DMF) which exhibit better performance compared to other methods on rating prediction in Section 4.5.1, and compare them with our proposed method HHMF on the top-k recommendation task. Figures 4, 5, 6, 7, 8 and 9 show the ranking accuracies of NeuMF, DMF and HGMF$_2$ and HHMF$_2$ when $d = 20$ (for other settings, the results are similar). From the results, we can conclude that HHMF consistently outperforms NeuMF, DMF and HGMF for different sizes (i.e., k) of the recommendation lists, which demonstrates that HHMF is superior to previous approaches also for the task of top-k recommendation.

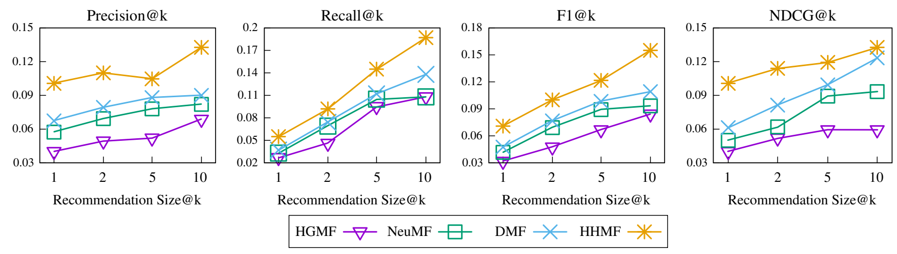


Fig. 4 Performance of top-k recommendation on FilmTrust

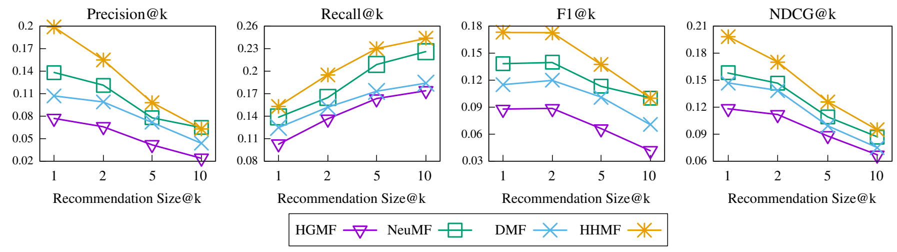


Fig. 5 Performance of top-k recommendation on Ciao

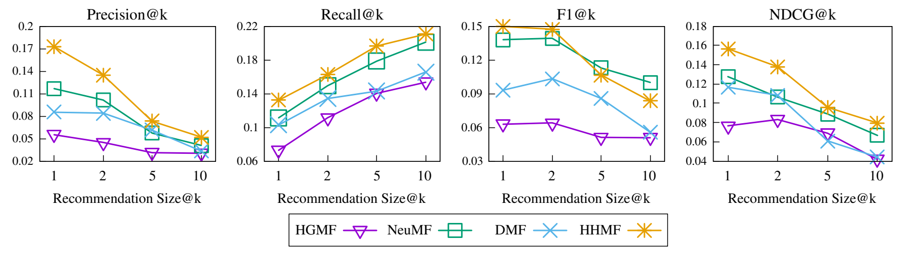


Fig. 6 Performance of top-k recommendation on Yelp

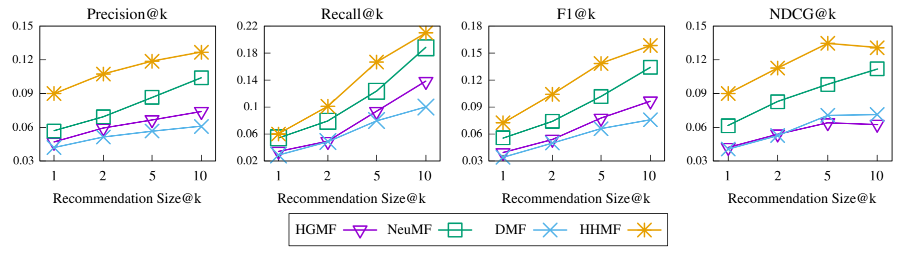


Fig. 7 Performance of top-k recommendation on Dianping

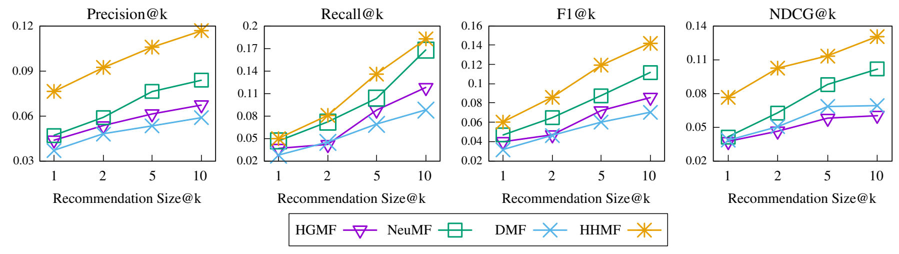


Fig. 8 Performance of top-k recommendation on MovieLens

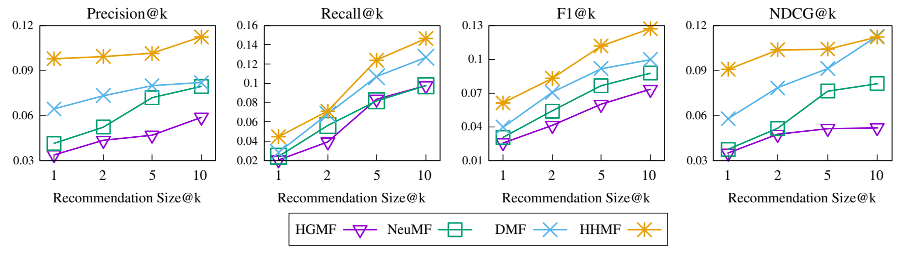


Fig. 9 Performance of top-k recommendation on Netflix

4.6 Sensitivity to Hyperparameters (Q2)

To answer question Q2, we report the performance of HHMF when different hyperparameters are used.

Impact of λ. Firstly, we evaluate the impact of the regularization weight λ. We compare HHMF with HGMF for the task of rating prediction, since HHMF and HGMF require the same hyperparameters and it is easy to illustrate the effect of changing one hyperparameter when the others are fixed. We set $d = 20$ and a two-level hierarchy is learned in HHMF and HGMF. The results using different λ are shown in Figure 10. From the results, we can draw the conclusion that the best λ values are consistent on these datasets (i.e., 0.01 or 0.1), although deviating from them would downgrade the performance.

Impact of group ratio. We also evaluate the influence of the group ratio on HHMF. We set $d = 20$ and use a two-level hierarchy. λ is tuned to be optimal for each specific value of the group ratio. The results regarding the rating prediction task are illustrated in Figure 11. Note that a group ratio of 10 achieves the best results. Larger or smaller group ratios do not help to improve the performance further.

Impact of d. Finally, we evaluate the performance when changing the latent dimensionality d. Figure 12 shows the performances of HHMF and HGMF when a two-level hierarchy is learned by the two methods. From Figure 12, we can conclude that increasing d can improve the performance of both HHMF and HGMF. On the other hand, HHMF consistently outperforms HGMF using the same value of d.

In summary, the best hyperparameters of HHMF on these datasets tend to be consistent as depicted in our extensive experiments, though deviating from these values will affect the results a lot. In practice, we recommend setting λ to be 0.01 or 0.1 and using a group ratio of 10 for HHMF.

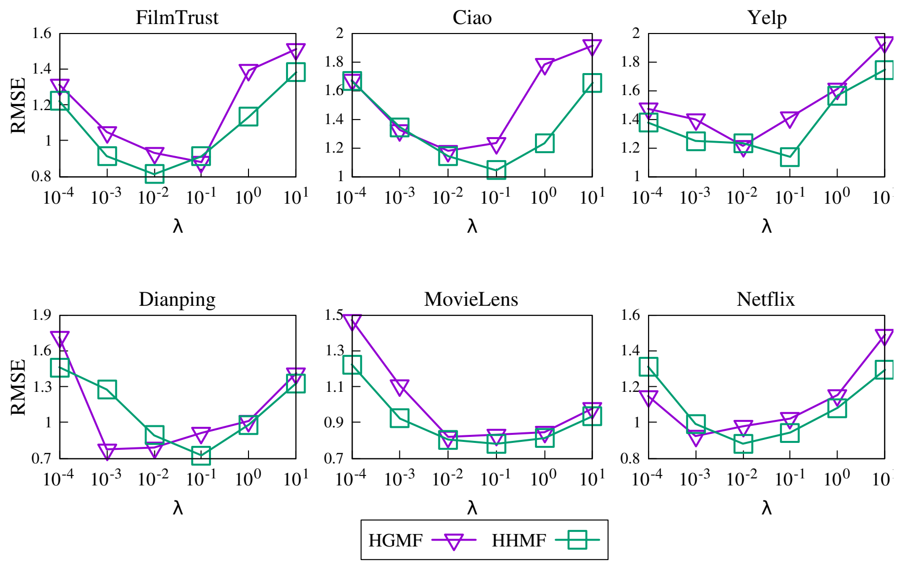


Fig. 10 Performance of rating prediction when using different values of λ

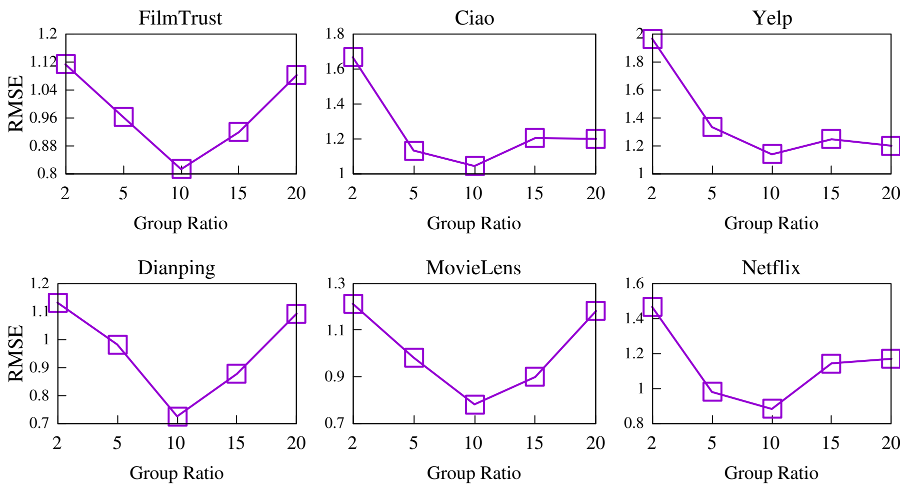


Fig. 11 Performance of rating prediction when using different values of group ratio

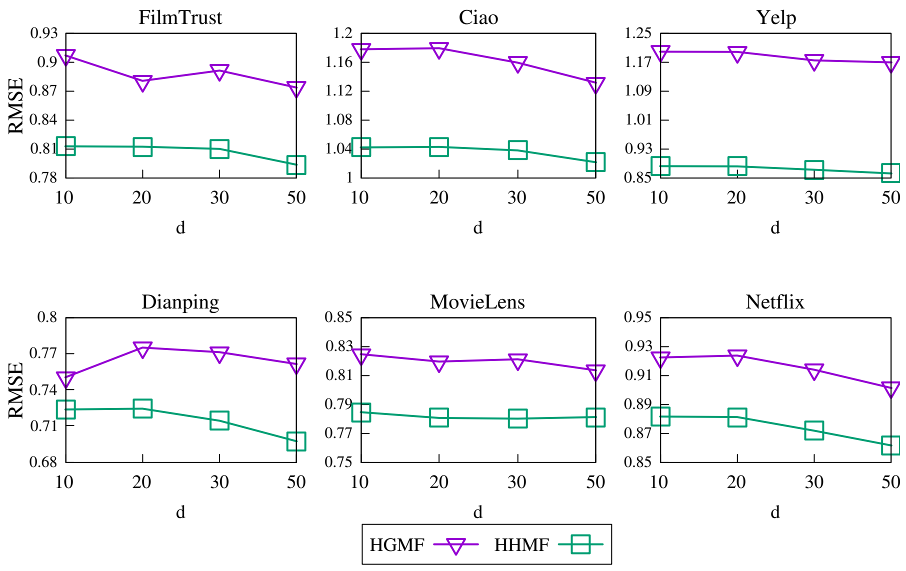


Fig. 12 Performance of rating prediction when using different values of d

4.7 Running Time (Q3)

In Section 4.5, we have demonstrated the superiority of HHMF over other models on the quality of recommendation. We now investigate the practicality of HHMF by assessing its runtime scalability. Note that the implementations of FISM, IHSR, DMF, NeuMF and NFM use different deep learning libraries and directly comparing the running time of Python implementations (e.g., NeuMF and NFM which utilize GPUs) with C++ implementations (e.g., CPU based methods like HGMF and HHMF) may not illustrate the scalability fairly. The experiments we show in this section are used to help readers understand that our proposed HHMF is both effective and efficient in practice, even when we do not use the best possible implementation and hardware.

We compare the running time of HHMF with that of HGMF and NFM. HGMF is the best CPU-based method as shown in Tables 3 and 4, while NFM is the most recent GPU-based baseline we have compared against in previous experiments. Our implementations of HGMF and HHMF adopt Intel® Math Kernel Library.[19] The main program which wraps Intel® Math Kernel Library is multithreading and implemented using standard C++ library. The implementation of NFM is provided by the authors of (He and Chua 2017) and implemented using TensorFlow. The experiments were conducted on a machine with Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz, 32 GB of main memory, Nvidia Titan Xp 12GB and Ubuntu 16.04. The programs of HGMF and HHMF are set to use 4 threads in our experiments.

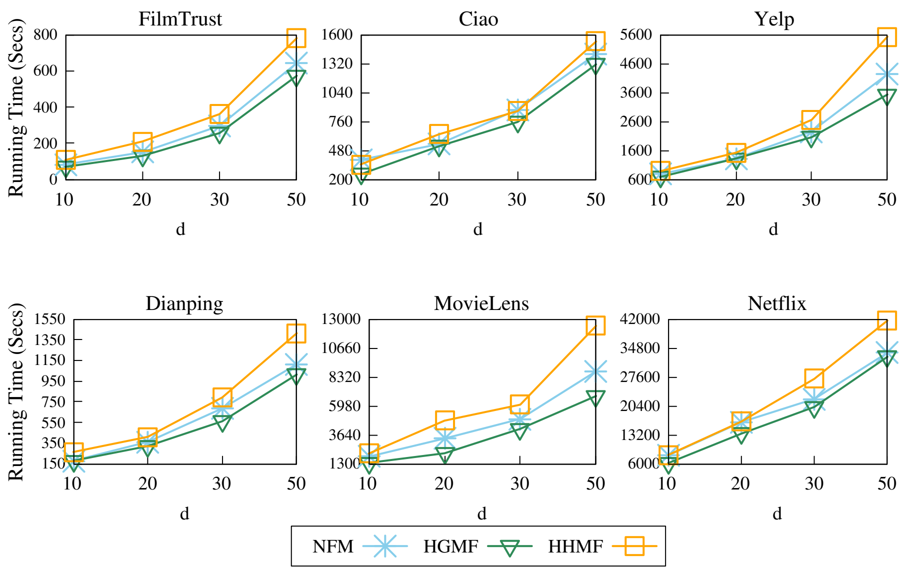


Fig. 13 Running time of NFM, HGMF and HHMF when using different values of d

Figure 13 shows the running time of NFM, HGMF and HHMF when a three-level hierarchy is learned and the group ratio is 10. For each test, we run each method once using its best hyperparameters (found in hyperparameter

[19] https://software.intel.com/en-us/mkl

search) and report their running time in this one-time experiment. We vary the value of latent dimensionality d in Figure 13. From Figure 13, we can see the cost of HHMF is typically higher than that of NFM and HGMF, using the same value of d. However, all the methods scale similarly and their runtime difference is not significant. This confirms our time complexity analysis in Section 3.4. Considering the recommendation quality of HHMF, we believe that the additional overhead pays off. Note that HGMF and HHMF can be deployed in a GPU environment when their implementation is modified using CUDA[20] and their running time is expected to drop further.

4.8 Analysis of the Learned 'Hidden' Structure (Q4)

HGMF and HHMF are designed to model the hierarchy of users and items, matching the natural hierarchical structures in the data. We now analyze the hidden hierarchies learned by HGMF and HHMF in order to assess whether they represent well the similarity among users/items and, in turn, help improve the quality of recommendation.

To measure the similarity between two user groups s and v, we adopt the *Pearson Correlation Coefficient* (PCC) which is widely used in recommender systems (Resnick et al. 1994):

$$\mathrm{Sim}_{sv} = \frac{\sum_{r\in R_{s*}\cap R_{v*}}(r-\bar{R}_{s*})\cdot(r-\bar{R}_{v*})}{\sqrt{\sum_{r\in R_{s*}\cap R_{v*}}(r-\bar{R}_{s*})^2}\sqrt{\sum_{r\in R_{s*}\cap R_{v*}}(r-\bar{R}_{v*})^2}}, \tag{15}$$

where $R_{s*}\cap R_{v*}$ indicates ratings of those items that both user groups have rated. $\bar{R}_{s*}$ and $\bar{R}_{v*}$ represent the average rating of user group s and user group v, respectively. We map PCC into range $[0,1]$ using function $f(x)=(x+1)/2$. The similarity between the two item groups is defined in a similar way.

For each user group, we compute the average PCC of all pairs of smaller user groups it contains (i.e., user groups from the lower level) for HGMF$_3$ and HHMF$_3$ when $d=20$ and the group ratio is 10. We report the average in-group PCC in Table 5. For item groups, we compute PCC similarly. In Table 5, 1→2 indicates the grouping from user/item groups at level 1 (where each group contains one user/item) to user/item groups at level 2, and 2→3 indicates the grouping from user/item groups at level 2 to user/item groups at level 3.

From Table 5, we can observe the following:

(1) The hierarchy learned in HHMF captures user/item grouping better than HGMF, since the average PCC of HHMF is higher than the corresponding PCC of HGMF in most cases.

(2) From lower to higher levels, the average PCC decreases. During rating prediction, the task is conducted top-bottom and thus the predicted ratings are refined from higher to lower levels. At higher levels, more user

[20] https://www.geforce.com/hardware/technology/cuda

Table 5 Average Pearson Correlation Coefficient in each level

Method	Grouping	FilmTrust	Ciao	Yelp	Dianping	MovieLens	Netflix
HGMF (user)	1→2	0.1021	0.3698	0.2411	0.3156	0.1711	0.2310
	2→3	0.0783	0.3012	0.2171	0.1865	0.1456	0.2101
HHMF (user)	1→2	0.1398	0.3912	0.2976	0.3981	0.1801	0.2481
	2→3	0.0911	0.3451	0.2511	0.2411	0.1611	0.2415
HGMF (item)	1→2	0.1312	0.3871	0.2731	0.3316	0.1187	0.2514
	2→3	0.1234	0.3229	0.2654	0.2341	0.1024	0.2431
HHMF (item)	1→2	0.1431	0.3812	0.3211	0.3612	0.1546	0.2741
	2→3	0.1287	0.3451	0.2899	0.2678	0.1347	0.2547

groups/item groups from the level below are included in one group, which introduces more noise.

To conclude, HHMF models better the user/item grouping structures compared to HGMF, which explains why HHMF outperforms HGMF (and other approaches that do not capture at all the hierarchical structures in the data).

5 Conclusion

In this paper, we proposed a novel hierarchical matrix factorization method called HHMF, which learns and uses the 'hidden' hierarchical structure in the user-item rating records to improve the quality of recommendation. HHMF can be applied in recommender systems, where the hierarchical structure is implicit. Our extensive experiments demonstrate that HHMF outperforms traditional MF methods, state-of-the-art hierarchical MF methods, and neural network-based methods.

Currently, there are a few limitations of HHMF. First, we only consider basic user-item rating information, instead of incorporating additional contextual information. We do this in order to demonstrate that the improvement achieved by HHMF over other methods is due to the effectiveness of our interdependent structure learning and rating prediction. Additionally, the number of layers in HHMF is a user-defined hyperparameter, as in other hierarchical MF approaches. Lastly, HHMF adopts *hard-assignment*, i.e., one user group or item group can only be affiliated to one topic group after the hierarchy is built, although the probabilities that one user group or item group is affiliated to different topic groups are also learned.

Towards addressing the aforementioned drawbacks, there are several possible directions for improving HHMF:

(1) In the future, we plan to incorporate additional contextual information (e.g., review text (Wang et al. 2018a; García-Durán et al. 2018), check-in records (Lu et al. 2017) and item metadata (Li et al. 2012)) into HHMF in order to further improve its performance.

(2) We will enhance the current optimization method and design an early-stop mechanism, such that HHMF does not require the hyperparameter of

layer number. This way, the algorithm will be able to stop before entering into deeper layers, if the current model converges.

(3) We will consider introducing additional soft-grouping mechanisms into HHMF such that each user group or item group can affiliate to more than one topic group. In this way, more flexibility is added to HHMF and its performance can potentially be improved.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Grant No. 61803249) and the Shanghai Sailing Program (Grant No. 18YF1407700).

References

Airoldi EM, Blei DM, Fienberg SE, Xing EP (2008) Mixed membership stochastic blockmodels. Journal of Machine Learning Research 9:1981–2014

Bell RM, Koren Y (2007) Lessons from the netflix prize challenge. SIGKDD Explorations 9(2):75–79

Bertsekas DP (2006) Nonlinear Programming. Athena Scientific

Beutel A, Murray K, Faloutsos C, Smola AJ (2014) Cobafi: Collaborative bayesian filtering. In: Proceedings of the 23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014, pp 97–108

Blei DM, Ng AY, Jordan MI (2003) Latent dirichlet allocation. Journal of Machine Learning Research 3:993–1022

Chen C, Zheng X, Wang Y, Hong F, Lin Z (2014) Context-aware collaborative topic regression with social matrix factorization for recommender systems. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence, AAAI '14, Québec City, Québec, Canada, July 27-31, 2014, pp 9–15

Dhillon IS (2001) Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, San Francisco, CA, USA, August 26-29, 2001, pp 269–274

Ding D, Li H, Huang Z, Mamoulis N (2017) Efficient fault-tolerant group recommendation using alpha-beta-core. In: Proceedings of the 26th ACM International Conference on Information and Knowledge Management, CIKM '17, Singapore, November 6-10, 2017, pp 2047–2050

Dror G, Koenigstein N, Koren Y, Weimer M (2012) The yahoo! music dataset and kdd-cup '11. In: Proceedings of KDD Cup 2011 competition, San Diego, CA, USA, 2011, vol 18, pp 8–18

Facebook (2015) Recommending items to more than a billion people. http://code.facebook.com/posts/861999383875667

Fan W, Wang H, Yu PS, Ma S (2003) Is random model better? on its accuracy and efficiency. In: Proceedings of the 3rd IEEE International Conference on Data Mining, ICDM '03, Melbourne, Florida, USA, December 19-22, 2003, pp 51–58

García-Durán A, Gonzalez R, Oñoro-Rubio D, Niepert M, Li H (2018) Transrev: Modeling reviews as translations from users to items. CoRR abs/1801.10095, URL http://arxiv.org/abs/1801.10095, 1801.10095

George T, Merugu S (2005) A scalable collaborative filtering framework based on co-clustering. In: Proceedings of the 5th IEEE International Conference on Data Mining, ICDM '05, Houston, Texas, USA, November 27-30, 2005, pp 625–628

Guo G, Zhang J, Yorke-Smith N (2013) A novel bayesian similarity measure for recommender systems. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence, IJCAI '13, Beijing, China, August 3-9, 2013, pp 2619–2625

Guo G, Zhang J, Thalmann D, Yorke-Smith N (2014) Etaf: An extended trust antecedents framework for trust prediction. In: Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM '14, Beijing, China, August 17-20, 2014, pp 540–547

Harper FM, Konstan JA (2016) The movielens datasets: History and context. ACM Transactions on Interactive Intelligent Systems 5(4):19:1–19:19

He R, McAuley J (2016) Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In: Proceedings of the 25th International World Wide Web Conference, WWW '16, Montreal, Canada, April 11-15, 2016, pp 507–517

He R, Lin C, Wang J, McAuley J (2016) Sherlock: Sparse hierarchical embeddings for visually-aware one-class collaborative filtering. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, IJCAI '16, New York, NY, USA, July 9-15, 2016, pp 3740–3746

He X, Chua T (2017) Neural factorization machines for sparse predictive analytics. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '17, Shinjuku, Tokyo, Japan, August 7-11, 2017, pp 355–364

He X, Liao L, Zhang H, Nie L, Hu X, Chua T (2017) Neural collaborative filtering. In: Proceedings of the 26th International World Wide Web Conference, WWW '17, Perth, Australia, April 3-7, 2017, pp 173–182

Herlocker JL, Konstan JA, Terveen LG, Riedl J (2004) Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems 22(1):5–53

Hu Y, Koren Y, Volinsky C (2008) Collaborative filtering for implicit feedback datasets. In: Proceedings of the 8th IEEE International Conference on Data Mining, ICDM '08, Pisa, Italy, December 15-19, 2008, pp 263–272

Jamali M, Huang T, Ester M (2011) A generalized stochastic block model for recommendation in social rating networks. In: Proceedings of the 5th ACM Conference on Recommender Systems, RecSys '11, Chicago, IL, USA, October 23-27, 2011, pp 53–60

Kabbur S, Ning X, Karypis G (2013) FISM: factored item similarity models for top-n recommender systems. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '13, Chicago, IL, USA, August 11-14, 2013, pp 659–667

Koenigstein N, Dror G, Koren Y (2011) Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In: Proceedings of the 5th ACM Conference on Recommender Systems, RecSys '11, Chicago, IL, USA, October 23-27, 2011, pp 165–172

Koren Y (2008) Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08, Las Vegas, Nevada, USA, August 24-27, 2008, pp 426–434

Koren Y, Bell RM, Volinsky C (2009) Matrix factorization techniques for recommender systems. IEEE Computer 42(8):30–37

Li H (2015) Social network based recommender systems. HKU Theses Online (HKUTO)

Li H (2018) Scalable and feasible learning and retrieval from matrix data. HKU Theses Online (HKUTO)

Li H, Cai F, Liao Z (2012) Content-based filtering recommendation algorithm using hmm. In: Proceedings of the 4th International Conference on Computational and Information Sciences, ICCIS '12, Chongqing, China, August 17-19, 2012., pp 275–277

Li H, Wu D, Mamoulis N (2014) A revisit to social network-based recommender systems. In: Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, Gold Coast, QLD, Australia, July 6-11, 2014, pp 1239–1242

Li H, Wu D, Tang W, Mamoulis N (2015) Overlapping community regularization for rating prediction in social recommender systems. In: Proceedings of the 9th ACM Conference on Recommender Systems, RecSys '15, Vienna, Austria, September 16-20, 2015, pp 27–34

Li H, Chan TN, Yiu ML, Mamoulis N (2017) FEXIPRO: fast and exact inner product retrieval in recommender systems. In: Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD '17, Chicago, IL, USA, May 14-19, 2017, pp 835–850

Li H, Liu Y, Mamoulis N, Rosenblum DS (2019) Translation-based sequential recommendation for complex users on sparse data. IEEE Transactions on Knowledge and Data Engineering

Lian D, Zhao C, Xie X, Sun G, Chen E, Rui Y (2014) Geomf: joint geographical modeling and matrix factorization for point-of-interest recommendation. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA, August 24-27, 2014, pp 831–840

Liu X, Aberer K (2013) Soco: a social network aided context-aware recommender system. In: Proceedings of the 22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013, pp 781–802

Lu Z, Li H, Mamoulis N, Cheung DW (2017) HBGG: a hierarchical bayesian geographical model for group recommendation. In: Proceedings of the 17th SIAM International Conference on Data Mining, SDM '17, Houston, Texas, USA, April 27-29, 2017., pp 372–380

Ma H, Yang H, Lyu MR, King I (2008) Sorec: social recommendation using probabilistic matrix factorization. In: Proceedings of the 17th ACM International Conference on Information and Knowledge Management, CIKM '08, Napa Valley, California, USA, October 26-30, 2008, pp 931–940

Ma H, Zhou D, Liu C, Lyu MR, King I (2011) Recommender systems with social regularization. In: Proceedings of the 4th International Conference on Web Search and Web Data Mining, WSDM '11, Hong Kong, China, February 9-12, 2011, pp 287–296

Maleszka M, Mianowska B, Nguyen NT (2013) A method for collaborative recommendation using knowledge integration tools and hierarchical structure of user profiles. Knowledge-Based Systems 47:1–13

Mashhoori A, Hashemi S (2012) Incorporating hierarchical information into the matrix factorization model for collaborative filtering. In: Proceedings of the 4th Asian Conference on Intelligent Information and Database Systems - Part III, ACIIDS '12, Kaohsiung, Taiwan, March 19-21, 2012, vol 7198, pp 504–513

Menon AK, Chitrapura KP, Garg S, Agarwal D, Kota N (2011) Response prediction using collaborative filtering with hierarchies and side-information. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '11, San Diego, CA, USA, August 21-24, 2011, pp 141–149

Ng AY, Jordan MI, Weiss Y (2001) On spectral clustering: Analysis and an algorithm. In: Proceedings of the 14th International Conference on Neural Information Processing Systems, NIPS '01, Vancouver, British Columbia, Canada, December 3-8, 2001, pp 849–856

Nikolakopoulos AN, Kouneli MA, Garofalakis JD (2015) Hierarchical itemspace rank: Exploiting hierarchy to alleviate sparsity in ranking-based recommendation. Neurocomputing 163:126–136

Nocedal J (1980) Updating quasi-newton matrices with limited storage. Mathematics of computation 35(151):773–782

Oentaryo RJ, Lim E, Low J, Lo D, Finegold M (2014) Predicting response in mobile advertising with hierarchical importance-aware factorization machine. In: Proceedings of the 7th ACM International Conference on Web Search and Data Mining, WSDM '14, New York, NY, USA, February 24-28, 2014, pp 123–132

Paterek A (2007) Improving regularized singular value decomposition for collaborative filtering. In: KDD cup and workshop, vol 2007, pp 5–8

Qian Y, Li H, Mamoulis N, Liu Y, Cheung DW (2017) Reverse k-ranks queries on large graphs. In: Proceedings of the 20th International Conference on Extending Database Technology, EDBT '17, Venice, Italy, March 21-24, 2017., pp 37–48

Rendle S (2010) Factorization machines. In: Proceedings of the 10th IEEE International Conference on Data Mining, ICDM '10, Sydney, Australia, December 14-17, 2010, pp 995–1000

Resnick P, Iacovou N, Suchak M, Bergstrom P, Riedl J (1994) Grouplens: An open architecture for collaborative filtering of netnews. In: Proceedings of the Conference on Computer Supported Cooperative Work, CSCW '94, Chapel Hill, NC, USA, October 22-26, 1994, pp 175–186

Salakhutdinov R, Mnih A (2007) Probabilistic matrix factorization. In: Proceedings of the 20th International Conference on Neural Information Processing Systems, NIPS '07, Vancouver, British Columbia, Canada, December 3-6, 2007, pp 1257–1264

Shan H, Kattge J, Reich PB, Banerjee A, Schrodt F, Reichstein M (2012) Gap filling in the plant kingdom - trait prediction using hierarchical probabilistic matrix factorization. In: Proceedings of the 29th International Conference on Machine Learning, ICML '12, Edinburgh, Scotland, UK, June 26 - July 1, 2012

Shi Y, Larson M, Hanjalic A (2014) Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. ACM Computing Surveys 47(1):3:1–3:45

Wang C, Blei DM (2011) Collaborative topic modeling for recommending scientific articles. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, San Diego, CA, USA, August 21-24, 2011, pp 448–456

Wang C, Niepert M, Li H (2018a) LRMM: learning to recommend with missing modalities. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP '18, Brussels, Belgium, October 31 - November 4, 2018, pp 3360–3370

Wang F, Li T, Wang X, Zhu S, Ding CHQ (2011) Community discovery using nonnegative matrix factorization. Data Mining and Knowledge Discovery 22(3):493–521

Wang Q, Cao Z, Xu J, Li H (2012) Group matrix factorization for scalable topic modeling. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '12, Portland, OR, USA, August 12-16, 2012, pp 375–384

Wang S, Tang J, Wang Y, Liu H (2015) Exploring implicit hierarchical structures for recommender systems. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence, IJCAI '15, Buenos Aires, Argentina, July 25-31, 2015, pp 1813–1819

Wang S, Du C, Zhao K, Li C, Li Y, Zheng Y, Wang Z, Chen H (2016) Random partition factorization machines for context-aware recommendations. In: Proceedings of the 17th International Conference on Web-Age Information Management - Part I, WAIM '16, Nanchang, China, June 3-5, 2016, vol 9658, pp 219–230

Wang S, Tang J, Wang Y, Liu H (2018b) Exploring hierarchical structures for recommender systems. IEEE Transactions on Knowledge and Data Engineering 30(6):1022–1035

Wang X, Pan W, Xu C (2014) HGMF: hierarchical group matrix factorization for collaborative recommendation. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM '14, Shanghai, China, November 3-7, 2014, pp 769–778

Xu W, Liu X, Gong Y (2003) Document clustering based on non-negative matrix factorization. In: Proceedings of the 26th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '03, Toronto, Canada, July 28 - August 1, 2003, pp 267–273

Xu Y, Lam W, Lin T (2014) Collaborative filtering incorporating review text and co-clusters of hidden user communities and item groups. In: Proceedings of the 23rd ACM International Conference on Information and Knowledge Management, CIKM '14, Shanghai, China, November 3-7, 2014, pp 251–260

Xue H, Dai X, Zhang J, Huang S, Chen J (2017) Deep matrix factorization models for recommender systems. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI '17, Melbourne, Australia, August 19-25, 2017, pp 3203–3209

Yang X, Steck H, Liu Y (2012) Circle-based recommendation in online social networks. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012, pp 1267–1275

Zhang S, Wang W, Ford J, Makedon F (2006) Learning from incomplete ratings using non-negative matrix factorization. In: Proceedings of the 6th SIAM International Conference on Data Mining, SDM '06, Bethesda, MD, USA, April 20-22, 2006, pp 549–553

Zhong E, Fan W, Yang Q (2012) Contextual collaborative filtering via hierarchical matrix factorization. In: Proceedings of the 12th SIAM International Conference on Data Mining, SDM '12, Anaheim, California, USA, April 26-28, 2012, pp 744–755

6 Appendix

6.1 Calculation of Equation 3

$$
\begin{aligned}
&Pr(Q, P, \theta, \mu, \phi, \kappa | R, \varXi) \\
&= Pr(Q, P | R, \varXi) \cdot Pr(\theta, \mu, \phi, \kappa | R, \varXi) \\
&= Pr(Q, P | R, \varXi) \cdot Pr(\theta, \mu, \phi, \kappa | R) \\
&= Pr(Q, P | R, \varXi) \cdot Pr(\theta, \kappa | R) \cdot Pr(\mu, \phi | R) \\
&= \frac{Pr(Q, P | R, \varXi) \cdot Pr(R|\theta, \kappa) \cdot Pr(R|\mu, \phi) \cdot Pr(\theta, \kappa) \cdot Pr(\mu, \phi)}{Pr(R) \cdot Pr(R)} \\
&\propto Pr(Q, P | R, \varXi) \cdot Pr(R|\theta, \kappa) \cdot Pr(R|\mu, \phi).
\end{aligned}
$$

6.2 Calculation of Equation 7

$$
\begin{aligned}
&Pr(\{Q^{(\ell)}\}_{\ell=1}^{l}, \{P^{(\ell)}\}_{\ell=1}^{l} \,|\, \{R^{(\ell)}\}_{\ell=1}^{l}, \{S^{(\ell)}\}_{\ell}^{l-1}, \{V^{(\ell)}\}_{\ell=1}^{l-1}, \varXi) \\
= &\frac{Pr(\{R^{(\ell)}\}_{\ell=1}^{l}, \{S^{(\ell)}\}_{\ell}^{l-1}, \{V^{(\ell)}\}_{\ell=1}^{l-1}, \varXi \,|\, \{Q^{(\ell)}\}_{\ell=1}^{l}, \{P^{(\ell)}\}_{\ell=1}^{l}) \cdot \mathrm{Pr}(\{Q^{(\ell)}\}_{\ell=1}^{l}, \{P^{(\ell)}\}_{\ell=1}^{l})}{Pr(\{R^{(\ell)}\}_{\ell=1}^{l}, \{S^{(\ell)}\}_{\ell}^{l-1}, \{V^{(\ell)}\}_{\ell=1}^{l-1}, \varXi)} \\
\propto &Pr(\{R^{(\ell)}\}_{\ell=1}^{l} \,|\, \{Q^{(\ell)}\}_{\ell=1}^{l}, \{P^{(\ell)}\}_{\ell=1}^{l}, \sigma_R^2) \cdot Pr(\{S^{(\ell-1)}\}_{\ell=1}^{l} \,|\, \{Q^{(\ell)}\}_{\ell=1}^{l}, \{P^{(\ell)}\}_{\ell=1}^{l}, \sigma_S^2) \\
&Pr(\{V^{(\ell-1)}\}_{\ell=1}^{l} \,|\, \{Q^{(\ell)}\}_{\ell=1}^{l}, \{P^{(\ell)}\}_{\ell=1}^{l}, \sigma_V^2) \cdot \mathrm{Pr}(\{Q^{(\ell)}\}_{\ell=1}^{l}) \cdot \mathrm{Pr}(\{P^{(\ell)}\}_{\ell=1}^{l}) \\
= &\prod_{\ell=1}^{l} \prod_{r_{ui} \in R^{(\ell)}} \mathcal{N}(\langle Q_u^{(\ell)}, P_i^{(\ell)} \rangle, \sigma_{R^{(\ell)}}^2) \cdot \prod_{\ell=1}^{l-1} \prod_{s_{ui} \in S^{(\ell)}} \mathcal{N}(\langle Q_u^{(\ell+1)}, P_i^{(\ell)} \rangle, \sigma_{S^{(\ell)}}^2) \\
&\prod_{\ell=1}^{l-1} \prod_{v_{ui} \in V^{(\ell)}} \mathcal{N}(\langle Q_i^{(\ell)}, P_j^{(\ell+1)} \rangle, \sigma_{V^{(\ell)}}^2) \cdot \prod_{\ell=1}^{l} \prod_{u=1}^{m^{(\ell)}} \phi_{e_u}^{(\ell+1)} \mathcal{N}(Q^{(\ell+1)}, \sigma_{Q^{(\ell+1)}}^2 \mathrm{I}) \\
&\prod_{\ell=1}^{l} \prod_{i=1}^{n^{(\ell)}} \kappa_{t_i}^{(\ell+1)} \mathcal{N}(P^{(\ell+1)}, \sigma_{P^{(\ell+1)}}^2 \mathrm{I}).
\end{aligned}
$$

6.3 Gradients used in Equation 9

$$\frac{\partial \mathcal{L}_{R^{(\ell)}}}{Q_u^{(\ell)}} = \nabla_q(R_{u\cdot}^{(\ell)}, \ell, \ell) - \mathcal{R}(Q_u^{(\ell)}) \tag{16}$$

$$\frac{\partial \mathcal{L}_{R^{(\ell)}}}{P_i^{(\ell)}} = \nabla_p(R_{\cdot i}^{(\ell)}, \ell, \ell) - \mathcal{R}(P_i^{(\ell)}) \tag{17}$$

$$\frac{\partial \mathcal{L}_{S^{(\ell-1)}}}{Q_u^{(\ell)}} = \nabla_q(S_{u\cdot}^{(\ell-1)}, \ell, \ell-1) - \mathcal{R}(Q_u^{(\ell)}) \tag{18}$$

$$\frac{\partial \mathcal{L}_{S^{(\ell-1)}}}{P_i^{(\ell-1)}} = \nabla_p(S_{\cdot i}^{(\ell-1)}, \ell, \ell-1) - \mathcal{R}(P_i^{(\ell-1)}) \tag{19}$$

$$\frac{\partial \mathcal{L}_{V^{(\ell-1)}}}{Q_u^{(\ell-1)}} = \nabla_q(V_{u\cdot}^{(\ell-1)}, \ell-1, \ell) - \mathcal{R}(Q_u^{(\ell-1)}) \tag{20}$$

$$\frac{\partial \mathcal{L}_{V^{(\ell-1)}}}{P_i^{(\ell)}} = \nabla_p(V_{\cdot i}^{(\ell-1)}, \ell-1, \ell) - \mathcal{R}(P_i^{(\ell)}) \tag{21}$$

where

$$\nabla_q(R_{u\cdot}^{(\ell)}, \ell_1, \ell_2) = \frac{1}{\sigma_R^2} \sum_{r_{ui} \in R^{(\ell)}} (r_{ui}^{(\ell)} - \langle Q_u^{(\ell_1)}, P_i^{(\ell_2)} \rangle) \cdot P_i^{(\ell)}$$

$$\nabla_p(R_{\cdot i}^{(\ell)}, \ell_1, \ell_2) = \frac{1}{\sigma_R^2} \sum_{r_{ui} \in R^{(\ell)}} (r_{ui}^{(\ell)} - \langle Q_u^{(\ell_1)}, P_i^{(\ell_2)} \rangle) \cdot Q_u^{(\ell)}$$

$$\nabla_q(S_{u\cdot}^{(\ell-1)}, \ell, \ell-1) = \frac{1}{\sigma_S^2} \sum_{r_{ui} \in R^{(\ell-1)}} \left(s_{ui}^{(\ell-1)} - \langle Q_u^{(\ell)}, P_i^{(\ell-1)} \rangle \right) \cdot P_i^{(\ell-1)}$$

$$\nabla_p(S_{\cdot i}^{(\ell-1)}, \ell, \ell-1) = \frac{1}{\sigma_S^2} \sum_{r_{ui} \in R^{(\ell)}} \left(s_{ui}^{(\ell-1)} - \langle Q_u^{(\ell)}, P_i^{(\ell-1)} \rangle \right) \cdot Q_u^{(\ell)}$$

$$\nabla_q(V_{u\cdot}^{(\ell-1)}, \ell-1, \ell) = \frac{1}{\sigma_V^2} \sum_{r_{ui} \in R^{(\ell)}} \left(v_{ui}^{(\ell-1)} - \langle Q_u^{(\ell-1)}, P_i^{(\ell)} \rangle \right) \cdot P_i^{(\ell)}$$

$$\nabla_p(V_{\cdot i}^{(\ell-1)}, \ell-1, \ell) = \frac{1}{\sigma_V^2} \sum_{r_{ui} \in R^{(\ell-1)}} \left(v_{ui}^{(\ell-1)} - \langle Q_u^{(\ell-1)}, P_i^{(\ell)} \rangle \right) \cdot Q_u^{(\ell-1)}$$

$$\mathcal{R}(Q_u^{(\ell)}) = \frac{1}{\sigma_Q^2} \sum_{r_{ui} \in R^{(\ell)}} \phi_{(e_u,u)}^{(\ell+1)} \Big[|WQ_u^{(\ell)}| (Q_u^{(\ell)} - \bar{Q}_{WQ_u^{(\ell)}}^{(\ell-1)}) + (Q_u^{(\ell)} - Q_{FQ_u^{(\ell)}}^{(\ell+1)}) \Big]$$

$$\mathcal{R}(P_i^{(\ell)}) = \frac{1}{\sigma_P^2} \sum_{r_{ui} \in R^{(\ell)}} \kappa_{(t_i,i)}^{(\ell+1)} \Big[|WP_i^{(\ell)}| (P_i^{(\ell)} - \bar{P}_{WP_i^{(\ell)}}^{(\ell-1)}) + (P_i^{(\ell)} - P_{FP_i^{(\ell)}}^{(\ell+1)}) \Big]$$