Probabilistic Factor Models for Web Site Recommendation

Hao Ma[*] , Chao Liu
Microsoft Research
One Microsoft Way
Redmond, WA 98052
haoma@microsoft.com
chaoliu@microsoft.com

Irwin King[†]
AT&T Labs Research, USA &
The Chinese University of
Hong Kong, Hong Kong
irwin@research.att.com
king@cse.cuhk.edu.hk

Michael R. Lyu
The Chinese University of
Hong Kong
Shatin, N.T., Hong Kong
lyu@cse.cuhk.edu.hk

ABSTRACT

Due to the prevalence of personalization and information filtering applications, modeling users' interests on the Web has become increasingly important during the past few years. In this paper, aiming at providing accurate personalized Web site recommendations for Web users, we propose a novel probabilistic factor model based on dimensionality reduction techniques. We also extend the proposed method to collective probabilistic factor modeling, which further improves model performance by incorporating heterogeneous data sources. The proposed method is general, and can be applied to not only Web site recommendations, but also a wide range of Web applications, including behavioral targeting, sponsored search, etc. The experimental analysis on Web site recommendation shows that our method outperforms other traditional recommendation approaches. Moreover, the complexity analysis indicates that our approach can be applied to very large datasets since it scales linearly with the number of observations.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Search Process, Information Filtering

General Terms

Algorithm, Experimentation

Keywords

Probabilistic Factor Modeling, Web Site Recommendation, Matrix Factorization, Recommender Systems

1. INTRODUCTION

How to find the most relevant search results for Web users has been extensively studied in both academia and industry during the past decade. Traditionally, commercial search engines try to satisfy online users' information needs based on the queries issued by users. By taking advantages of the advanced learning and ranking algorithms, commercial search engine services have already become indispensable in solving Web users' various information needs.

However, due to the rapid growth of information on the Web, especially for the social Web, the traditional search paradigm can no longer meet Web users' ever-growing information needs. The Web is now shifting online users from "search" to "discovery". The current paradigm is that a user who has information needs or questions will search on commercial search engines for related Web sites or answers. Different from "search", the concept of "discovery" refers to the ability to push relevant content to users either based on the meta data that have already been collected about the users, or by exposing their activities that their friends and social networks are engaged in.

From the above definitions, we can see that these two concepts are different but closely related since essentially, the idea of "discovery" is similar to "autonomous search", which helps users search for things when they do not even yet know what they want. Due to its personalized nature, "discovery" or "autonomous search", which combines a person's context with others' past opinions and actions, can therefore create tremendous value and relevance.

In this paper, in order to enhance online users' experience, we investigate a novel application, Web site recommendation, to help users proactively discover relevant Web sites by conducting autonomous search based on their past online behaviors.

Many methods [7, 13, 17, 19, 22, 23, 24, 32] have been proposed to tackle recommendation or collaborative filtering problems in the literature. Typically, in traditional recommender systems, modeling users' interests needs to utilize a user-item rating matrix, which explicitly specifies users' preferences. However, in the scenario of Web site recommendation, it is infeasible to ask Web users to explicitly rate Web sites they like or dislike. Instead, we can only take advantages of implicit user behavior data in the past to make recommendations. Hence, most of the traditional collaborative filtering approaches cannot be directly applied to the Web site recommendation task.

In this work, aiming at solving the Web site recommendation problem effectively and efficiently, we propose a proba-

[*] This work was partially done when the first author was on summer internship at Microsoft Research.

[†] Irwin King is currently on leave from CUHK to be with AT&T Labs.

SIGIR'11, July 24–28, 2011, Beijing, China.
Copyright 2011 ACM 978-1-4503-0757-4/11/07 ...$10.00.

bilistic factor modeling framework by utilizing implicit user-site frequency data based on the following motivations:

1. A Web user's preference on different Web sites can be represented by how frequently a user visits each Web site;

2. Higher visiting frequency on a site from a user means heavy information need on this site while lower frequency indicates less interest;

3. User-query issuing frequency data can be incorporated to refine a user's preference since normally the user will first issue a query before clicking a Web site.

The first two motivations lead to a Probabilistic Factor Model (PFM) based on dimensionality reduction techniques. More specifically, the user-site frequency matrix is factorized into two low-dimensional factor matrices: user latent matrix and site latent matrix. The value of each dimension in a user factor vector follows a gamma distribution, which indicates a user's preference in a latent topic. A site factor vector also has similar definition. Then, the observed user-site frequency matrix is assumed to follow Poisson distribution with the mean generated by the inner product of the user latent matrix and the item latent matrix. Finally, an efficient multiplicative updating algorithm is proposed to learn the latent factor matrices.

By considering the third motivation, we design a Collective Probabilistic Factor Model (CPFM) which simultaneously factorizes user-site matrix and user-query matrix by sharing the same user latent space. The experimental analysis shows that CPFM model can further improve the recommendation quality.

The remainder of this paper is structured as follows. In Section 2, we provide an overview of several major approaches for recommender systems and some related work. Section 3 details the concept of low-rank matrix factorization. The proposed probabilistic framework and the learning algorithm are presented in Section 4. The results of an empirical analysis is presented in Section 5, followed by the conclusion and future work in Section 6.

2. RELATED WORK

In this section, we review two research topics which are relevant to our work: Web user interests prediction and recommender systems.

2.1 Web User Interests Prediction

In this subsection, we review some closely related tasks in the literature to predict Web user interests.

Personalized search engine is a natural application to Web user interests modeling, which improves the ranking system (ranking problem can also be considered as recommendation problem) by incorporating an individual's historical activities. In [27], Sun et al. conducted sophisticated analysis on the correlation between users, their queries and search results clicked to model user preferences. The experimental results indicate that the proposed CubeSVD approach can significantly improve Web search performance. In [20], Qiu et al. studied how a search engine can learn a user's preference automatically based on her past click history and how it can use the user preference to personalize search results. The experiments show that personalized search based on user preference yields significant improvements over the best existing ranking mechanism in the literature. Teevan et al. in [28] also utilized implicit information about users' interests to re-rank Web search results within a relevance feedback framework.

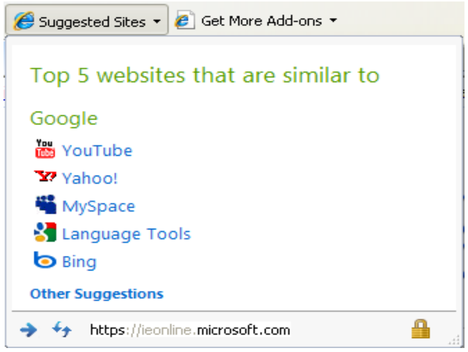


Figure 1: IE8 Site Suggestion for "Google.com"

Applications of implicit feedback to predict user future visits are also studied in [6, 11]. These systems typically establish historical click trails of a user or a community of users, and assess the accuracy of statistical machine learning models which predict future page visits [29].

Recently, White et al. evaluated how to effectively predict user interests based on five contextual information sources: social, historic, task, collection and user interaction [29]. The user's Web page visits are automatically assigned to Open Directory Project (ODP) categories and then used to predict future visits of sites within a category. Different from this approach, in the work presented in this paper, we focus on predicting a lower granularity of actual frequencies of the visits, as opposed to a category level.

Starting from the release of Internet Explorer 8 (IE8), an important feature was introduced - "Suggested Site". As shown in Figure 1, IE8 lists Top 5 Web sites that are similar to the site a user types in the address bar. So far, this feature makes Web site recommendations based on a single site, which does not include any personalized results. In this paper, we will study how to personalize the suggestion results based on users' past online behaviors.

2.2 Recommender Systems

Our work is closely related to the research of recommender systems. In this subsection, we review several major approaches for recommender systems, especially for collaborative filtering.

A number of algorithms have been proposed to improve both the recommendation quality and the scalability problems. As discussed in [9], one of the most commonly-used and successfully-deployed recommendation approaches is collaborative filtering. These collaborative filtering algorithms can be divided into two main categories: neighborhood-based and model-based approaches. Neighborhood-based methods, or memory-based methods, mainly concentrate on finding the similar users [2, 10] or items [5, 17, 24] for generating recommendations. The neighborhood-based recommendation algorithms are based on the assumption that those who agreed in the past tend to agree again in the future. They usually fall into two classes: user-based approaches and item-based approaches. User-based approaches predict the ratings of active users based on the ratings of similar users found, while item-based approaches predict the ratings of active users based on the computed information of items similar to those chosen by the active user.

In addition to the neighborhood-based approaches, the model-based approaches to collaborative filtering train a predefined model by employing the observed user-item ratings. Algorithms in this category include the clustering model [12], the aspect models [7, 8, 25], the latent factor model [3], the Bayesian hierarchical model [31] and the ranking model [18]. Recently, due to its efficiency in dealing with large datasets, several low-dimensional matrix approximation methods [1, 13, 21, 22, 23, 26] have been proposed. These methods all focus on fitting the user-item rating matrix using low-rank approximations, and employ the matrix to make further predictions. The low-rank matrix factorization methods are very efficient in training since they assume that in the user-item rating matrix, only a small number of factors influence preferences, and that a user's preference vector is determined by how each factor applies to that user. Low-rank matrix approximations based on minimizing the sum-squared errors can be easily solved using Singular Value Decomposition (SVD). In [22], Salakhutdinov et al. proposed a probabilistic graphic model by assuming some Gaussian observation noises on observed user-item ratings. The proposed model achieved promising prediction results. In their following work proposed in [23], the Gaussian-Wishart priors are placed on the user and item hyper-parameters. Since exact inference is intractable in the new model, a Gibbs sampling method is proposed to iteratively learn the user and item latent matrices.

All of the low-dimensional approximation methods mentioned above are based on factorizing the user-item rating matrix. Normally, in a recommender system, users can assign 5-scale integer ratings (from 1 to 5, and 5 indicates "love" while 1 means "dislike") to items[1]. However, in the case of Web site recommendation, the range of the user-site frequency data is much wider than the rating data. A user can visit a site only once, but probably will also visit another site thousands of times or even more. The wide range of the frequency data poses new challenges in modeling the Web site recommendation problem. Some state-of-the-art algorithms based on the rating data may not perform well on the frequency data. For example, in the method proposed in [22], the rating data are modeled as Gaussian distributions with some noises. This assumption is not suitable for modeling the frequency data since the underlying distribution is probably not Gaussian. Hence, most of previous low-dimensional approximation methods cannot generate good recommendation results when they are applied to the Web site recommendation problems.

3. MATRIX FACTORIZATION IN RECOMMENDER SYSTEMS

As mentioned in Section 2, a popular method in recommender systems is to factorize the user-item rating matrix, and to utilize the factorized user-specific and item-specific matrices for further missing data prediction [22, 23, 30]. The premise behind a low-dimensional factor model is that there are only a small number of factors influencing the preferences, and that a user's preference vector is determined by how each factor applies to that user [21].

In recommender systems, considering an $m \times n$ user-item rating matrix R describing m users' numerical ratings on n items, a low-rank matrix factorization approach seeks to approximate the rating matrix R by a multiplication of d-rank factors $R \approx UV^T$, where $U \in \mathbb{R}^{m\times d}$ and $V \in \mathbb{R}^{n\times d}$ with $d << \min(m, n)$. The matrix R in the real-world is usually very sparse since most of the users only rates few items.

The Singular Value Decomposition (SVD) method is employed to estimate a matrix R by minimizing

$$\min_{U,V} \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - U_i V_j^T)^2, \qquad (1)$$

where U_i and V_j are row vectors with d values, I_{ij} is the indicator function that is equal to 1 if user i rated item j and equal to 0 otherwise.

Another popular method in recommender systems is the Probabilistic Matrix Factorization (PMF) method proposed in [22]. The conditional distribution over the observed ratings is defined as:

$$p(R|U, V, \sigma_R^2) = \prod_{i=1}^{m}\prod_{j=1}^{n}\left[\mathcal{N}\left(R_{ij}|U_i V_j^T, \sigma_R^2\right)\right]^{I_{ij}}, \qquad (2)$$

where $\mathcal{N}(x|\mu, \sigma^2)$ is the probability density function of the Gaussian distribution with mean μ and variance σ^2. The zero-mean spherical Gaussian priors are also placed on user and item feature vectors:

$$p(U|\sigma_U^2)=\prod_{i=1}^{m}\mathcal{N}(U_i|0, \sigma_U^2\mathbf{I}),\ p(V|\sigma_V^2)=\prod_{j=1}^{n}\mathcal{N}(V_j|0, \sigma_V^2\mathbf{I}). \quad (3)$$

Hence, through a Bayesian inference, we have the following objective function:

$$\min_{U,V} \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} I_{ij}(R_{ij} - U_i V_j^T)^2 + \frac{\lambda_1}{2}||U||_F^2 + \frac{\lambda_2}{2}||V||_F^2, \quad (4)$$

where $\lambda_1, \lambda_2 > 0$. The optimization problem is to minimize the sum-of-squared-errors objective function with quadratic regularization terms. Gradient based approaches can be applied to find a local minimum. The above algorithm is perhaps one of the most popular methods in collaborative filtering.

Actually, when modeling the observed rating data, both SVD and PMF have the underlying assumption about Gaussian distribution. This assumption may not be appropriate when working on the frequency data. Hence, these models may encounter problems in generating good recommendation results. The experimental analysis conducted in Section 5 demonstrates our concerns. In next section, we propose two probabilistic models which can potentially generate better Web site recommendation results than SVD and PMF.

4. RECOMMENDATION FRAMEWORK

4.1 Problem Definition

The major task studied in this paper is to predict how frequently a user will visit a Web site. In commercial search engines or browser toolbars, query logs record the activities

[1] Different recommender systems adopt different rating scales. Some systems may use 10-scale integer ratings.

Table 1: Samples of Search Engine Query Logs

ID	Query	URL
358	facebook	http://www.facebook.com
358	rww	http://www.readwriteweb.com
3968	iphone4	http://www.apple.com
3968	ipad	http://www.apple.com
...	...	...

		Web sites					
		v_1	v_2	v_3	v_4	v_5	v_6
Web users	u_1		68	1		15	
	u_2	42			13		24
	u_3		72	12		11	2
	u_4	15			33		
	u_5		85	45			63

(a) User-Site Matrix

		Queries				
		z_1	z_2	z_3	z_4	z_5
Web users	u_1	12		5	6	
	u_2		23		5	1
	u_3		14		35	18
	u_4	25		11	4	
	u_5		12	5		24

(b) User-Query Matrix

Figure 2: Aggregated Data From Query Logs

of Web users, which reflect their interests and the relationships between users and queries, as well as users and clicked Web sites. As shown in Table 1, each line of the data we need includes the following information: a user ID, a query issued by the user, and a Web site on which the user clicked.

By aggregating the data in Table 1, we can construct two matrices, as shown in Figure 2. Figure 2(a) is the user-site frequency matrix, which contains how many times a user visited a Web site. Figure 2(b) gives the user-query frequency matrix, which indicates how frequently a user issued a query.

The problem under investigation is essentially how to effectively and efficiently predict the missing values of the user-site frequency matrix by employing these data sources. To attack this problem, in Section 4.2, we propose a novel probabilistic factor model by using only the user-site frequency matrix, while in Section 4.3, we introduce a collective probabilistic factor model by utilizing both user-site and user-query frequency matrices, which can further improve the model performance.

4.2 Probabilistic Factor Model

Our proposed Probabilistic Factor Model (PFM) is a generative probabilistic model, which can be represented by the graphical model in Figure 3. Let F be an $m \times n$ data matrix whose element f_{ij} is the observed count of event (or feature, or item) j by user i. Y is a matrix of expected counts with the same dimensions as F, and y_{ij} denotes an element in matrix Y. Every observed element f_{ij} in matrix F is assumed to follow Poisson distribution with the mean y_{ij} in matrix Y, respectively. The matrix Y is factorized into two matrices U and V, where U is an $m \times d$ matrix, V is an $n \times d$ matrix and d is the dimensionality. Each element u_{ik} $(k = 1, ..., d)$ in U encodes the preference of user i to latent topic k, and each v_{jk} in V can be interpreted as the affinity of site j to the latent topic k. Finally, u_{ik} and v_{jk} are given the Gamma distributions as the empirical priors.

There are two reasons we use Gamma distributions to model u_{ik} and v_{jk} instead of Gaussian or other distributions: (1) Gamma distribution is suitable for modeling non-negative values, while Gaussian distribution can model both negative and non-negative values. If we allow negative values in u_{ik} and v_{jk}, potentially, the model will generate negative frequency values, which is unreasonable in the real world. (2) The Gamma distribution is already proved to be effective in modeling document latent vectors in text analysis [4], where the document-word relation is also represented as a frequency matrix.

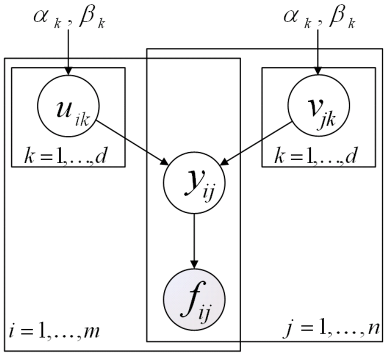


Figure 3: Graphical Model for Probabilistic Factor Modeling

Therefore, the generative process of an observed user-site count f_{ij} in our model follows:

1. Generate $u_{ik} \sim \text{Gamma}(\alpha_k, \beta_k), \forall k$.
2. Generate $v_{jk} \sim \text{Gamma}(\alpha_k, \beta_k), \forall k$.
3. Generate y_{ij} occurrences of item or event j from user i with outcome $y_{ij} = \sum_{k=1}^{d} u_{ik} v_{jk}$.
4. Generate $f_{ij} \sim \text{Poisson}(y_{ij})$.

The gamma distributions of U and V follow the probabilistic functions

$$p(U|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \prod_{i=1}^{m} \prod_{k=1}^{d} \frac{u_{ik}^{\alpha_k - 1} \exp(-u_{ik}/\beta_k)}{\beta_k^{\alpha_k} \Gamma(\alpha_k)}, \quad (5)$$

$$p(V|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \prod_{j=1}^{n} \prod_{k=1}^{d} \frac{v_{jk}^{\alpha_k - 1} \exp(-v_{jk}/\beta_k)}{\beta_k^{\alpha_k} \Gamma(\alpha_k)}, \quad (6)$$

where $\boldsymbol{\alpha} = \{\alpha_1, ..., \alpha_d\}$, $\boldsymbol{\beta} = \{\beta_1, ..., \beta_d\}$, $u_{ik} > 0$, $v_{jk} > 0$, $\alpha_k > 0$ and $\beta_k > 0$, $\Gamma(\cdot)$ is the Gamma function.

The Poisson distribution of F given Y can then be defined as

$$p(F|Y) = \prod_{i=1}^{m} \prod_{j=1}^{n} \frac{y_{ij}^{f_{ij}} \exp(-y_{ij})}{f_{ij}!}, \quad (7)$$

where $y_{ij} = \sum_{k=1}^{d} u_{ik} v_{jk}$.

Since $Y = UV^T$, the posterior distribution of U and V given F can be modeled as

$$p(U, V|F, \boldsymbol{\alpha}, \boldsymbol{\beta}) \propto p(F|Y) p(U|\boldsymbol{\alpha}, \boldsymbol{\beta}) p(V|\boldsymbol{\alpha}, \boldsymbol{\beta}). \quad (8)$$

Hence, we can infer the log of the posterior distribution $p(U, V|F, \boldsymbol{\alpha}, \boldsymbol{\beta})$ as follows

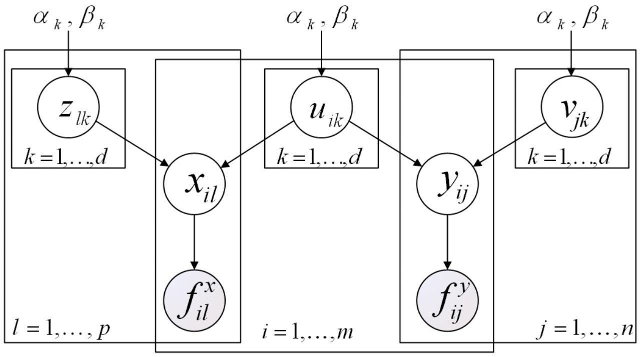


Figure 4: Graphical Model for Collective Probabilistic Factor Modeling

$$
\begin{aligned}
\mathcal{L}(U,V;F) \quad = \quad & \sum_{i=1}^{m}\sum_{k=1}^{d}((\alpha_k-1)\ln(u_{ik}/\beta_k)-u_{ik}/\beta_k) \\
& +\sum_{j=1}^{n}\sum_{k=1}^{d}((\alpha_k-1)\ln(v_{jk}/\beta_k)-v_{jk}/\beta_k) \\
& +\sum_{i=1}^{m}\sum_{j=1}^{n}(f_{ij}\ln y_{ij}-y_{ij})+\text{const.} \qquad (9)
\end{aligned}
$$

Taking derivatives on $\mathcal{L}$ with respect to u_{ik} and v_{jk}, we have

$$
\begin{aligned}
\frac{\partial\mathcal{L}}{\partial u_{ik}} &= \sum_{j=1}^{n}(f_{ij}v_{jk}/y_{ij}-v_{jk})+(\alpha_k-1)/u_{ik}-1/\beta_k, \\
\frac{\partial\mathcal{L}}{\partial v_{jk}} &= \sum_{i=1}^{m}(f_{ij}u_{ik}/y_{ij}-u_{ik})+(\alpha_k-1)/v_{jk}-1/\beta_k. \quad (10)
\end{aligned}
$$

Using similar techniques proposed by Lee and Seung in [16], by setting the learning rates to

$$
\frac{u_{ik}}{\sum_{j=1}^{n}v_{jk}+1/\beta_k} \text{ and } \frac{v_{jk}}{\sum_{i=1}^{m}u_{ik}+1/\beta_k}
$$

respectively, we can obtain the following multiplicative updating rules:

$$
\begin{aligned}
u_{ik} &\leftarrow u_{ik}\frac{\sum_{j=1}^{n}(f_{ij}v_{jk}/y_{ij})+(\alpha_k-1)/u_{ik}}{\sum_{j=1}^{n}v_{jk}+1/\beta_k}, \\
v_{jk} &\leftarrow v_{jk}\frac{\sum_{i=1}^{m}(f_{ij}u_{ik}/y_{ij})+(\alpha_k-1)/v_{jk}}{\sum_{i=1}^{m}u_{ik}+1/\beta_k}. \qquad (11)
\end{aligned}
$$

4.3 Collective Probabilistic Factor Model

In many Web applications, we have multiple data sources instead of a single data source. In the case of Web site recommendation, the Web users not only browsed many Web sites, but also issued several queries. The queries issued by users can also represent the interests of the users. If we can incorporate this information, we can potentially improve the model performance.

Actually, our proposed model in Section 4.2 can be easily extended to incorporate heterogeneous data sources. When we are observing the example matrices in Figure 2(a) and Figure 2(b), we notice that these two data sources share the same user space. Hence, we propose a Collective Probabilistic Factor Model (CPFM) by sharing the same user latent space. Intuitively, CPFM method should outperform PFM method since the additional user-query matrix can help model users' preferences more accurately. Figure 4 shows the graphical model for mining these two data matrices.

In this graphical model, F^x is an $m\times p$ data matrix whose element f^x_{il} represents how many times user i issued query l. X is a matrix of expected counts with the same dimensions as F^x, and x_{il} denotes an element in matrix X. Every observed element f^x_{il} in matrix F^x is assumed to follow Poisson distribution with the mean x_{il} in matrix X, respectively. The matrix X is factorized into two matrices U and Z, where Z is a $p\times d$ matrix. Each element z_{lk} in Z encodes the affinity of query l to the latent topic k. The definitions of F^y, Y, U, V, u_{ik} and v_{jk} are the same as the definitions in Section 4.2.

Similar to Eq. (8), we have the posterior distribution of U, V and Z, given F^x and F^y:

$$
\begin{aligned}
& p(U,V,Z|F^x,F^y,\boldsymbol{\alpha},\boldsymbol{\beta}) \\
& \quad \propto p(F^x|X)p(F^y|Y)p(U|\boldsymbol{\alpha},\boldsymbol{\beta})p(V|\boldsymbol{\alpha},\boldsymbol{\beta})p(Z|\boldsymbol{\alpha},\boldsymbol{\beta}).(12)
\end{aligned}
$$

In this equation, $p(F^x|X)$ and $p(F^y|Y)$ are defined as:

$$
p(F^x|X)=\prod_{i=1}^{m}\prod_{l=1}^{p}\frac{x_{il}^{f^x_{il}}\exp(-x_{il})}{f^x_{il}!}, \qquad (13)
$$

$$
p(F^y|Y)=\prod_{i=1}^{m}\prod_{j=1}^{n}\frac{y_{ij}^{f^y_{ij}}\exp(-y_{ij})}{f^y_{ij}!}, \qquad (14)
$$

where $x_{il}=\sum_{k=1}^{d}u_{ik}z_{lk}$ and $y_{ij}=\sum_{k=1}^{d}u_{ik}v_{jk}$.

Moreover, since every element in Z also follows a Gamma distribution, similar to $p(U|\boldsymbol{\alpha},\boldsymbol{\beta})$ and $p(V|\boldsymbol{\alpha},\boldsymbol{\beta})$, $p(Z|\boldsymbol{\alpha},\boldsymbol{\beta})$ can be defined as:

$$
p(Z|\boldsymbol{\alpha},\boldsymbol{\beta})=\prod_{l=1}^{p}\prod_{k=1}^{d}\frac{z_{lk}^{\alpha_k-1}\exp(-z_{lk}/\beta_k)}{\beta_k^{\alpha_k}\Gamma(\alpha_k)}. \qquad (15)
$$

Hence, by inferring Eq. (12), we have the log of the posterior distribution for CPFM model:

$$
\begin{aligned}
& \mathcal{L}(U,V,Z;F^x,F^y) \\
& \quad =\sum_{i=1}^{m}\sum_{l=1}^{p}(f^x_{il}\ln x_{il}-x_{il})+\sum_{i=1}^{m}\sum_{j=1}^{n}(f^y_{ij}\ln y_{ij}-y_{ij}) \\
& \quad +\sum_{i=1}^{m}\sum_{k=1}^{d}((\alpha_k-1)\ln(u_{ik}/\beta_k)-u_{ik}/\beta_k) \\
& \quad +\sum_{j=1}^{n}\sum_{k=1}^{d}((\alpha_k-1)\ln(v_{jk}/\beta_k)-v_{jk}/\beta_k) \\
& \quad +\sum_{l=1}^{p}\sum_{k=1}^{d}((\alpha_k-1)\ln(z_{lk}/\beta_k)-z_{lk}/\beta_k)+\text{const.}\,(16)
\end{aligned}
$$

Similar to PFM, we can easily obtain the multiplicative updating rules for learning U, V and Z:

$$
\begin{aligned}
u_{ik} &\leftarrow u_{ik} \frac{\displaystyle\sum_{j=1}^{n}(f^{y}_{ij}v_{jk}/y_{ij}) + \sum_{l=1}^{p}(f^{x}_{il}z_{lk}/x_{il}) + (\alpha_k - 1)/u_{ik}}{\displaystyle\sum_{j=1}^{n} v_{jk} + \sum_{l=1}^{p} z_{lk} + 1/\beta_k}, \\
v_{jk} &\leftarrow v_{jk} \frac{\sum_{i=1}^{m}(f^{y}_{ij}u_{ik}/y_{ij}) + (\alpha_k - 1)/v_{jk}}{\sum_{i=1}^{m} u_{ik} + 1/\beta_k}, \\
z_{lk} &\leftarrow z_{lk} \frac{\sum_{i=1}^{m}(f^{x}_{il}u_{ik}/x_{il}) + (\alpha_k - 1)/z_{lk}}{\sum_{i=1}^{m} u_{ik} + 1/\beta_k}. \qquad (17)
\end{aligned}
$$

In Eq. (17), we treat the information from the user-site matrix and the user-query matrix equally. However, sometimes we may want to control how much information to use in each side, and to find a balance between these two data sources. Hence, we add a smoothing parameter θ in order to tune the importance of these two data sources. The updating rules for u_{ik} in Eq. (17) is then changed to

$$
u_{ik} \leftarrow u_{ik} \frac{\displaystyle\theta\sum_{j=1}^{n}(f^{y}_{ij}v_{jk}/y_{ij}) + (1-\theta)\sum_{l=1}^{p}(f^{x}_{il}z_{lk}/x_{il}) + (\alpha_k - 1)/u_{ik}}{\displaystyle\theta\sum_{j=1}^{n} v_{jk} + (1-\theta)\sum_{l=1}^{p} z_{lk} + 1/\beta_k}. \qquad (18)
$$

Our proposed method is general, and can be easily extended to incorporate other contextual information for other research problems. In this section, we utilize the user-query information to improve the PFM method. Actually, in the query logs, every Web site is also associated with some queries. Hence, a natural extension is to further incorporate the site-query frequency matrix by adding another matrix factorization in Figure 4. We hope this can help learn the site latent vectors more accurately. Besides Web site recommendation, many other research problems also have similar frequency data, like sponsored search, behavioral targeting, text mining, etc. Our model can also be easily applied to these problems. We do not discuss the details in this paper since our focus of this paper is to illustrate how to make Web site recommendations.

4.4 Complexity Analysis

The main computation of gradient methods is evaluating the object function $\mathcal{L}$ and its gradients against variables. In PFM model, because of the sparsity of matrix F, the computational complexity of evaluating the object function $\mathcal{L}$ is $O(N_F d)$, where N_F is the number of nonzero entries in matrix F. The computational complexities for gradients $\frac{\partial \mathcal{L}}{\partial U}$ and $\frac{\partial \mathcal{L}}{\partial V}$ in Eq. (10) are also $O(N_F d)$. Therefore, the total computational complexity in one iteration is $O(N_F d)$, which indicates that the computational time of our PFM method is linear with respect to the number of observations in the user-site frequency matrix. The total complexity of PFM is then $O(N_F dr)$, where r is the number of iterations. Similarly, for CPFM, the complexity is $O(N_{F^x} dr + N_{F^y} dr)$, where N_{F^x} and N_{F^y} are the number of nonzero entries in user-query and user-site matrices, respectively. Since our algorithm will converge after 10 to 20 iterations, this complexity analysis shows that our proposed approach is very efficient and can scale up with respect to very large datasets.

Table 2: Statistics of User-Site and User-Query Frequency Matrices

Statistics	User-Site Frequency	User-Query Frequency
Min. Num.	4	10
Max. Num.	9,969	4,693
Avg. Num.	20.33	23.05

5. EXPERIMENTAL ANALYSIS

In this section, we conduct several experiments to compare the recommendation quality of our PFM and CPFM approaches with other baseline and state-of-the-art collaborative filtering or recommendation methods.

Our experiments are intended to address the following questions:

1. How does our approach compare with the baseline and published state-of-the-art algorithms?
2. How do the model parameters α_k and β_k affect the accuracy of prediction?
3. What is the performance difference when we use different dimensions to represent users, sites and queries?
4. How does the smooth parameter θ affect the prediction quality in CPFM?
5. Is our algorithm efficient for large datasets?

5.1 Metrics

We use two metrics, the Normalized Mean Absolute Error (NMAE) and the Normalized Root Mean Square Error (NRMSE), to measure the prediction accuracy.

The metric NMAE is defined as:

$$
NMAE = \frac{\sum_{i,j} |(f_{i,j} - \widehat{f}_{i,j})/f_{i,j}|}{N}, \qquad (19)
$$

where $f_{i,j}$ denotes the frequency user i visited Web site j, $\widehat{f}_{i,j}$ denotes the frequency user i visited Web site j as predicted by a method, and N denotes the number of tested data. The metric NRMSE is defined as:

$$
NRMSE = \sqrt{\frac{\sum_{i,j}((f_{i,j} - \widehat{f}_{i,j})/f_{i,j})^2}{N}}. \qquad (20)
$$

From the definitions, we can see that a smaller NMAE or NRMSE value means a better performance.

5.2 Dataset

The primary source of data for this work is the anonymized logs of Web sites visited by users who opted-in to provide data through a widely-distributed browser toolbar. Each line of the data source may contain a lot of information. We only keep those entries which include a unique identifier for the user, query issued by the user, and the Web sites visited by the user. Similar to the work in [29], in order to remove variability caused by geographic and linguistic variations in search behaviors, we only include those entries generated in the English speaking United States locale.

The experiments conducted in this paper are based on one month browser toolbar data, which represents billions of Web site visits. In order to remove some outliers and

Table 3: Performance Comparison (Dimensionality = 10)

Training Data	Metrics	UserMean	SiteMean	SVD	PMF	NMF	GaP	PFM	CPFM
90%	NMAE	2.246	1.094	0.488	0.476	0.465	0.440	0.432	**0.427**
	Improve	80.98%	60.96%	12.50%	10.29%	8.17%	2.95%		
	NRMSE	3.522	2.171	0.581	0.570	0.554	0.532	0.529	**0.520**
	Improve	85.24%	76.05%	10.50%	8.77%	6.14%	2.26%		
80%	NMAE	2.252	1.096	0.490	0.478	0.468	0.441	0.434	**0.428**
	Improve	80.99%	60.95%	12.65%	10.46%	8.55%	2.95%		
	NRMSE	3.714	2.159	0.584	0.571	0.560	0.533	0.530	**0.520**
	Improve	86.00%	75.91%	10.96%	8.93%	7.14%	2.44%		

Table 4: Performance Comparison (Dimensionality = 20)

Training Data	Metrics	UserMean	SiteMean	SVD	PMF	NMF	GaP	PFM	CPFM
90%	NMAE	2.246	1.094	0.469	0.460	0.449	0.426	0.413	**0.409**
	Improve	81.79%	62.61%	12.79%	11.09%	8.91%	3.99%		
	NRMSE	3.522	2.171	0.568	0.556	0.542	0.521	0.503	**0.496**
	Improve	85.92%	77.15%	12.68%	10.79%	8.49%	4.80%		
80%	NMAE	2.252	1.096	0.470	0.462	0.451	0.427	0.415	**0.410**
	Improve	81.79%	62.59%	12.77%	11.26%	9.09%	3.98%		
	NRMSE	3.714	2.159	0.570	0.558	0.545	0.522	0.504	**0.498**
	Improve	86.59%	76.93%	12.63%	10.75%	8.62%	4.60%		

clean up the data, we require that every user should at least visited 10 Web sites, and each Web site should at least be visited by 10 users. Moreover, we also constrain that a user should visit each Web site at least four times. URLs of all the Web sites are truncated to the site level. After pruning, we have 165,403 unique users, 265,367 unique URLs and 442,598 unique queries. Totally, the pruned dataset records 53,089,262 Web site visits. In the constructed user-site frequency matrix, 2,612,016 entries are observed, while in the user-query frequency matrix, the number of observed entries is 833,581. Some other statistics are shown in Table 2. From this table, we can see that a user visited a site for 9,969 times in a month, which shows that the range of user-site frequency matrix is very wide.

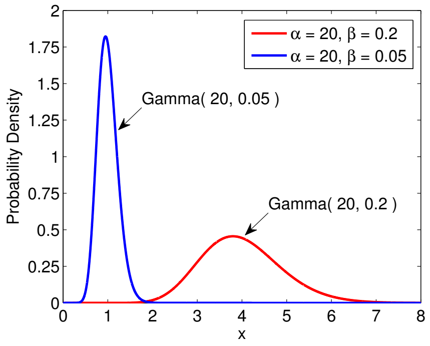


Figure 5: Gamma Distributions

5.3 Comparison

In this section, in order to show the effectiveness of our proposed recommendation approaches, we compare the recommendation results of the following methods:

1. **UserMean**: this is a baseline method, which uses the mean frequency value of every user to predict the missing values.

2. **SiteMean**: this is also a baseline method, which utilizes the mean frequency value of every site to predict the missing values.

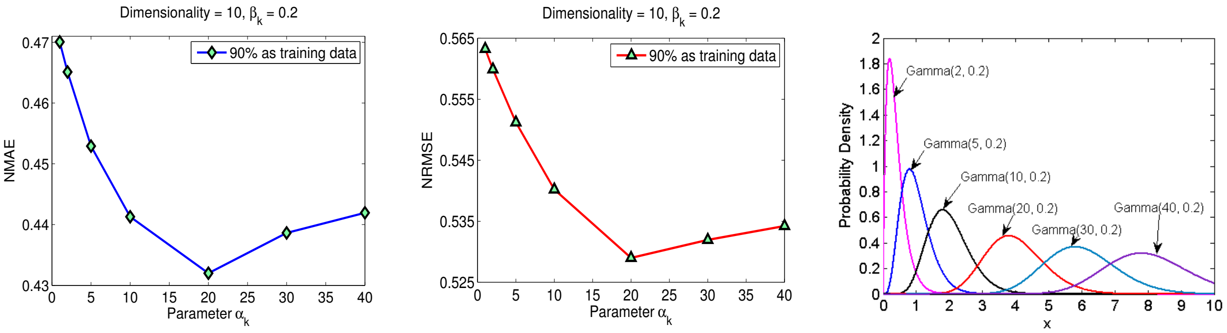
3. **SVD**: this is a well-known method in matrix factorization. We have shown the details of this method in Section 3. It is also been utilized in large scale collaborative filtering tasks [14].

4. **PMF**: this method is proposed by Salakhutdinov and Minh in [22]. The details of this method are also introduced in Section 3.

5. **NMF**: this method is originally proposed in [15, 16] for image analysis. However, it is widely used in collaborative filtering recently. The underlying distribution is also modeled as Poisson distribution.

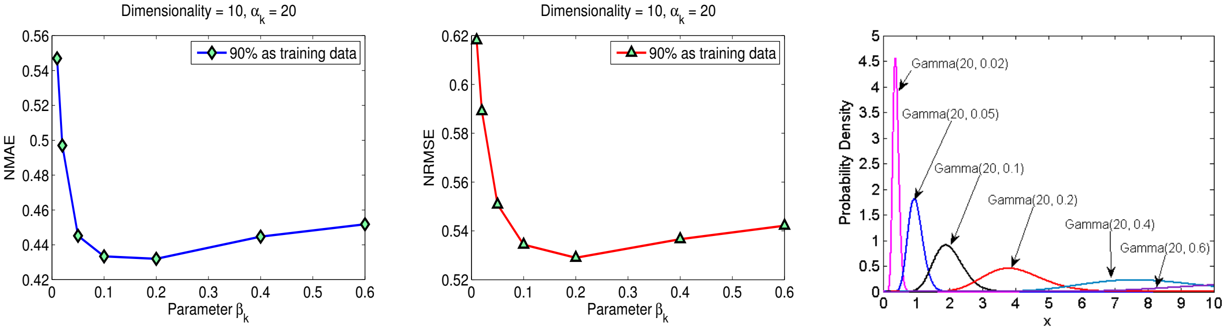
6. **GaP**: this approach is proposed in [4], and it is originally used to model text mining problems. The documents are modeled as a mixture of Gamma distributions.

We use different amounts of training data (90%, 80%) to test the algorithms. Training data 90%, for example, means we randomly select 90% of the observed data as the training data to predict the remaining 10%. The random selection was carried out 5 times independently, and we report the average results. The experimental results using 10 and 20 dimensions to represent the latent features are shown in Table 3 and Table 4, respectively. The standard deviations of the results generated by our methods are all around 0.005. In Table 3, we set all $\alpha_k = 20$, $\beta_k = 0.2$ in our PFM and CPFM models, while in Table 4, we change the settings to $\alpha_k = 20$, $\beta_k = 0.05$ for both PFM and CPFM. In Section 5.4, we will explain why we need to choose different α_k and β_k values when the dimensions are different. In addition, in



(a) NMAE (b) NRMSE (c) Gamma Distributions

Figure 6: Impact of Parameter α_k in PFM



(a) NMAE (b) NRMSE (c) Gamma Distributions

Figure 7: Impact of Parameter β_k in PFM

all the experiments, we set the parameter $\theta = 0.1$ for our CPFM method.

From the results, we can observe that our methods consistently outperform other approaches in all the settings, which shows the promising future of our methods. The percentages in Table 3 and Table 4 are the improvements of our CPFM method over the corresponding approaches.

We observe that the two baseline methods, UserMean and SiteMean, have the worst performance. Among these two methods, SiteMean performs much better than UserMean. This may indicate that users generally have diverse visiting patterns than sites. We also notice that two Gaussian based methods SVD and PMF are not suitable for factorizing frequency data, which coincides with our discussion in Section 2.

5.4 Impact of Dimensionality

From Table 3 and Table 4, we can see that our methods perform better when we choose a larger dimensionality. This is reasonable since more dimensions give us more flexibility to represent both user and site latent vectors. However, this is not always true since in the case we choose a very large dimensionality, we may experience severe overfitting problems.

As we discussed above, we set different α_k and β_k values when choosing different dimensions. This is because we model $\sum_{k=1}^{d} u_{ik}v_{jk}$ as the mean of a Poisson distribution. Since the frequencies are fixed, if we use a larger dimensionality, we need to employ some smaller values of u_{ik} and v_{jk}. That is why in the case of $d = 10$, the optimal parameter settings are $\alpha_k = 20$ and $\beta_k = 0.2$, while in the case where $d = 20$, we set $\alpha_k = 20$ and $\beta_k = 0.05$. We also plot the Gamma probability density functions for these two settings in Figure 5. From this figure, we can see that we will have a very high probability to generate smaller values if we choose the setting $\beta_k = 0.05$. This observation can help guide us to efficiently select appropriate parameters once the dimensionality is decided.

5.5 Impact of Parameters α_k and β_k

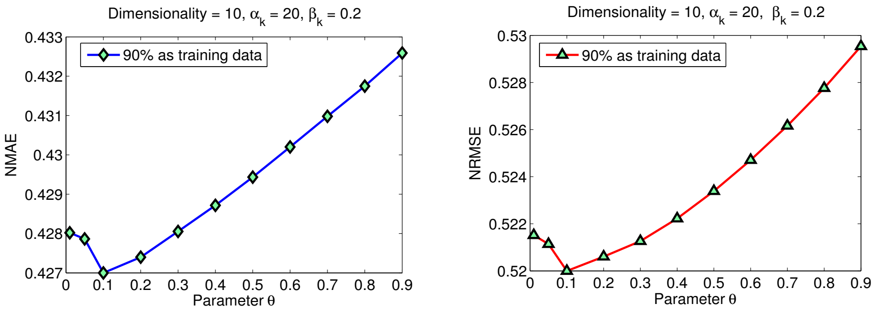
Parameters α_k and β_k are two important parameters since they define the shapes and scales of the Gamma distributions. In Figure 6 and Figure 7, we independently study how these two parameters affect the model performance in PFM. Figure 6 shows the impact of parameter α_k, given $\beta_k = 0.2$. We can see that the model achieves the best performance when $\alpha_k = 20$. A larger or smaller α_k value will lead to hurt the model performance.

Figure 6(c) shows some related Gamma distributions with β_k fixed. From this figure, we can easily interpret why different α_k and β_k values can significantly influence the performance. When $\alpha_k = 20$ and $\beta_k = 0.2$, the model has the best performance. At the same time, this Gamma distribution will generate values from 3 to 5 with high probabilities, which also means that these values can best represent user and site latent vectors. However, other parameter settings generate either much smaller or much larger values, which will potentially hurt the model performance.

The impact of parameter β_k is shown in Figure 7, and it shares similar trends as shown in Figure 6.

5.6 Impact of Parameter θ

In our CPFM method proposed in this paper, the parameter θ balances the information from the user-site matrix and



(a) NMAE

(b) NRMSE

Figure 8: Impact of Parameter θ in CPFM

Table 5: Efficiency Analysis (90% as Training Data)

PFM				
Dimensions	10D	20D	40D	60D
Seconds per Iteration	6.06	14.73	32.67	53.87
CPFM				
Dimensions	10D	20D	40D	60D
Seconds per Iteration	8.77	20.36	46.13	78.29

the user-query matrix. It controls how much our method should use the information in each matrix. If $\theta = 1$, we only factorize the user-site frequency matrix, and simply employ users' site visiting pattern in learning. If θ equals to a very small value, we will count more on information from the user-query frequency matrix to learn users' interests. For other cases in-between, we fuse information from the user-site matrix and the user-query matrix for collective probabilistic factor modeling and, furthermore, to make Web site recommendation for the users.

The impacts of θ on NMAE and NRMSE are shown in Figure 8. We observe that the value of θ impacts the recommendation results a lot, which illustrates that fusing these two data sources can greatly improve the recommendation quality. As θ increases, the NMAE and NRMSE decrease at first, but when θ surpasses a certain threshold, the NMAE and NRMSE increase with further increase of the value of θ. This phenomenon confirms with our initial intuition that fusing these two data sources together can generate better performance than purely using the user-site frequency matrix for recommendations.

In Figure 8(a) and Figure 8(b), when using 90% as training data, we observe that, our CPFM method achieves the best performance when θ is around 0.1, while smaller values like $\theta = 0.01$ or larger values like $\theta = 0.3$ can potentially hurt the model performance. Also a smaller θ is preferred to a larger θ indicates that we need to use more about user-query information than user-site information. The reason is perhaps the training data of user-site matrix is very sparse, which is less effective in learning the accurate interests of users.

5.7 Efficiency Analysis

The complexity analysis indicates that our method is linear with the observations. We also conduct some experiments on measuring the efficiency of our method. The Seconds-per-Iteration evaluation results are shown in Table 5 for both PFM and CPFM methods. We find that our methods are very efficient since our method will converge after 10 to 20 iterations. Hence, in 10-dimension case, we only need around 2 minutes to learn our PFM and CPFM models. CPFM is generally a little bit slower than PFM since CPFM needs to deal with one extra user-query matrix.

All the experiments are conducted on a workstation containing an *Intel Xeon* CPU (2.5 *GHz*) with 8G memory.

6. CONCLUSION AND FUTURE WORK

In this paper, aiming at providing personalized, accurate and efficient Web site recommendations for Web users, we propose two novel probabilistic factor models: PFM and CPFM. The experimental analysis on the large browser toolbar data shows the effectiveness and efficiency of our methods. The complexity analysis indicates our methods can be applied to very large datesets since they are linear with the observations in the user-site and user-query matrices.

In order to reduce the model complexity, in our PFM and CPFM methods, we use the same set of α_k and β_k values for both user, site and query latent features. Actually, since users, sites and queries are totally different objects, they may belong to completely different Gamma distributions. Hence, in order to further improve the recommendation performance, in the future, we can slightly modify the model, and allow setting different α_k and β_k values for these three objects.

In all the experiments conducted in this paper, we need to set the values for α_k and β_k. In the future, we will investigate a more intelligent algorithm to automatically determine these values. We may run into some challenges since the data is very sparse, hence, learning α_k and β_k automatically may experience overfitting problems. Another solution is that we can further assume a distribution on top of these two parameters. However, this is also difficult to implement since Gamma distribution does not have a very good conjugate prior distribution.

In our experiment, we only measure how accurate we can predict the user-site visiting frequency. Sometimes, only being accurate cannot meet the application requirements. We also plan to design a prototype for users to use, and ask users to measure the effectiveness of the proposed methods.

For the PFM model itself, we can make further assumption that every user or site is generated from a mixture of Gamma distributions. We can also assume that the mixture coefficients follow a Dirichlet distribution. Since we probably cannot infer the exact form of this model, we need to use sampling or variational approximation methods to solve the problem. This may however increase the complexity of the model.

Our proposed frame work is general; hence, in the future, we plan to apply our methods to other research problems, such as behavioral targeting and sponsored search. We can use the proposed methods to predict how many times a user will click an advertisement. Finally, we can also apply our methods to many text mining tasks, including clustering, classification, retrieval tasks, etc.

Acknowledgments

The work described in this paper was partially supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK 415210 and Project No. CUHK 415410), and partially supported by a Google Focused Grant Project under "Mobile 2014". The authors also would like to thank the reviewers for their helpful comments.

7. REFERENCES

[1] R. Bell, Y. Koren, and C. Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *Proc. of KDD '07*, pages 95–104, San Jose, California, USA, 2007.

[2] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of UAI '98*, 1998.

[3] J. Canny. Collaborative filtering with privacy via factor analysis. In *Proc. of SIGIR '02*, pages 238–245, Tampere, Finland, 2002.

[4] J. Canny. GaP: a factor model for discrete data. In *Proc. of SIGIR '04*, pages 122–129, Sheffield, United Kingdom, 2004.

[5] M. Deshpande and G. Karypis. Item-based top-n recommendation. *ACM Transactions on Information Systems*, 22(1):143–177, 2004.

[6] S. Gündüz and M. T. Özsu. Recommendation models for user accesses to web pages. In *Proc. of ICANN/ICONIP'03*, pages 1003–1010, Istanbul, Turkey, 2003.

[7] T. Hofmann. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *Proc. of SIGIR '03*, pages 259–266, Toronto, Canada, 2003.

[8] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems*, 22(1):89–115, 2004.

[9] Z. Huang, H. Chen, and D. Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):116–142, 2004.

[10] R. Jin, J. Y. Chai, and L. Si. An automatic weighting scheme for collaborative filtering. In *Proc. of SIGIR '04*, pages 337–344, Sheffield, United Kingdom, 2004.

[11] F. Khalil, J. Li, and H. Wang. Integrating recommendation models for improved web page prediction accuracy. In *Proc. of ACSC '08*, pages 91–100, Wollongong, Australia, 2008.

[12] A. Kohrs and B. Merialdo. Clustering for collaborative filtering applications. In *Proceedings of CIMCA*, 1999.

[13] Y. Koren. Collaborative filtering with temporal dynamics. In *Proc. of KDD '09*, pages 447–456, Paris, France, 2009.

[14] M. Kurucz, A. A. Benczur, and K. Csalogany. Methods for large scale svd with missing values. In *Proceedings of KDD Cup and Workshop*, 2007.

[15] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[16] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Proc. of NIPS '00*, pages 556–562, 2000.

[17] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, pages 76–80, Jan/Feb 2003.

[18] N. N. Liu and Q. Yang. Eigenrank: a ranking-oriented approach to collaborative filtering. In *Proc. of SIGIR '08*, pages 83–90, Singapore, Singapore, 2008.

[19] H. Ma, I. King, and M. R. Lyu. Effective missing data prediction for collaborative filtering. In *Proc. of SIGIR '07*, pages 39–46, Amsterdam, The Netherlands, 2007.

[20] F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In *Proc. of WWW '06*, pages 727–736, Edinburgh, Scotland, 2006.

[21] J. D. M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proc. of ICML '05*, pages 713–719, Bonn, Germany, 2005.

[22] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Proc. of NIPS '07*, 2007.

[23] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proc. of ICML '08*, 2008.

[24] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proc. of WWW '01*, pages 285–295, Hong Kong, Hong Kong, 2001.

[25] L. Si and R. Jin. Flexible mixture model for collaborative filtering. In *Proc. of ICML '03*, 2003.

[26] N. Srebro and T. Jaakkola. Weighted low-rank approximations. In *Proc. of ICML '03*, pages 720–727, Washington, DC, USA, 2003.

[27] J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen. Cubesvd: a novel approach to personalized web search. In *Proc. of WWW '05*, pages 382–390, Chiba, Japan, 2005.

[28] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *Proc. of SIGIR '05*, pages 449–456, Salvador, Brazil, 2005.

[29] R. W. White, P. Bailey, and L. Chen. Predicting user interests from contextual information. In *Proc. of SIGIR '09*, pages 363–370, Boston, MA, USA, 2009.

[30] K. Yu, S. Zhu, J. Lafferty, and Y. Gong. Fast nonparametric matrix factorization for large-scale collaborative filtering. In *Proc. of SIGIR '09*, pages 211–218, Boston, MA, USA, 2009.

[31] Y. Zhang and J. Koren. Efficient bayesian hierarchical user modeling for recommendation system. In *Proc. of SIGIR '07*, pages 47–54, Amsterdam, The Netherlands, 2007.

[32] D. Zhou, S. Zhu, K. Yu, X. Song, B. L. Tseng, H. Zha, and C. L. Giles. Learning multiple graphs for document recommendations. In *Proc. of WWW '08*, Beijing, China, 2008.