4-2014

Latent factor transition for dynamic collaborative filtering

Chengyi ZHANG
Simon Fraser University

Ke WANG
Simon Fraser University

Hongkun YU

Jianling SUN

Ee Peng LIM
Singapore Management University, eplim@smu.edu.sg

Latent Factor Transition for Dynamic Collaborative Filtering

Chenyi Zhang[*†] Ke Wang[†§] Hongkun Yu[*†] Jianling Sun[*] Ee-Peng Lim[‡]

Abstract

User preferences change over time and capturing such changes is essential for developing accurate recommender systems. Despite its importance, only a few works in collaborative filtering have addressed this issue. In this paper, we consider evolving preferences and we model user dynamics by introducing and learning a transition matrix for each user's latent vectors between consecutive time windows. Intuitively, the transition matrix for a user summarizes the time-invariant pattern of the evolution for the user. We first extend the conventional probabilistic matrix factorization and then improve upon this solution through its fully Bayesian model. These solutions take advantage of the model complexity and scalability of conventional Bayesian matrix factorization, yet adapt dynamically to user's evolving preferences. We evaluate the effectiveness of these solutions through empirical studies on six large-scale real life data sets.

Keywords: Latent factor transition; Preference evolving; Dynamic recommendation

1 Introduction

A founding principle of collaborative filtering is that if two users share similar interests on some items, they also likely share similar interests on other items. This simple preference propagation model from one item to another is challenged when user behaviors change over time. For example, a user rated cartoon movies at earlier years, then action movies some years later, and romantic movies more recently, and for another user, such a path of changes may be different. Another changing scenario voiced in [9] is that user's expertise level may upgrade from amateur to connoisseur over time. In both cases, the exact change in preferences depends on the user because such changes are a reflection of user's life experiences. As a result, even though two users rated cartoon movies similarly some years ago, they may rate

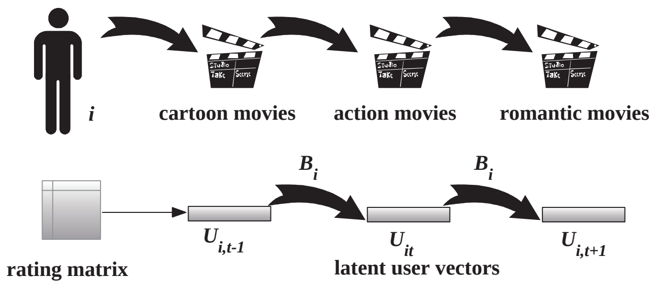


Figure 1: Evolution of user i's preferences

the same movie differently or like very different kinds of movies at a later time. In this situation, it is largely irrelevant to predict current preferences based on the similarity of preferences a few years ago.

On the other hand, it is possible that, for other users, the similarity of past preferences remains a good indicator of the similarity of current preferences. Therefore, it does not work to simply partition the rating data by time and learn a model using the data in each partition. This method not only misses the dependence of preferences for some users across time windows, but also accelerates the well known data sparsity problem. In addition, as pointed out in [6], temporal dynamics in recommendation are different from concept drift studied in machine learning [16]. In the case of recommendation, the evolution of each user's behaviors is determined by a potentially different set of factors, and there is no single global concept to be learnt.

1.1 Our Contributions

We assume that user preferences evolve gradually: the preference of user i at time t depends on the preference of the user at time $t-1$. Such temporal dependence is the basic idea of many statistical dynamic approaches such as hidden Markov model [3] and Kalman filter [5]. We model the temporal dependence for each user i through a $D \times D$ *transition matrix* B_i, where D is the dimensionality in the latent space: the latent vector U_{it} of user i at time t is a linear combination, specified by the rows of B_i, of the user's latent vector $U_{i,t-1}$ at time $t-1$, that is, U_{it} has the mean $B_i U_{i,t-1}$. This relationship is illustrated in Fig. 1. Intuitively, B_i captures the time-invariant pattern of the evolution for user i. For example, if user i increas-

[*]College of Computer Science, Zhejiang Univeristy
[†]School of Computing Science, Simon Fraser University
[‡]School of Information Systems, Singapore Management University
[§]Work done while visiting the Living Analytics Research Centre, Singapore Management University

ingly prefers the movies directed by James Cameron over time, the entry (j, j) in B_i will have a value larger than 1, assuming that the jth latent factor corresponds to James Cameron. The conventional static model can be treated as the special case of having the identity transition matrix B_i. Learning the transition matrices that help predict unknown ratings in the next time point is the main task in this paper.

The contributions of this paper are as follows: (1) We propose temporal probabilistic matrix factorization (TMF) and its fully Bayesian treatment model (BTMF), by incorporating a transition matrix into the conventional matrix factorization methods. This approach provides a clean solution by capturing temporal dynamics through the transition matrix and leveraging the principled Bayesian matrix factorization methodology. (2) We present gradient descent and MCMC to infer the parameters and transition matrices of these two models. (3) We conduct extensive experiments on six large-scale data sets. The empirical results demonstrate appealing improvements over the conventional matrix factorization and the state-of-the-art time-aware methods.

Although predicting the future rating is the focus in this paper, the learnt transition matrices have other applications. For example, since the transition matrix for a user captures the time-invariant aspect of user's evolution patterns, we can group users using learnt transition matrices as the features and develop a customized recommendation strategy for each group. A further investigation of this topic is beyond the scope of this paper.

1.2 Related Work

The latent factor model [7, 8], especially matrix factorization [12, 13, 14, 20], has been extensively and effectively used in collaborative filtering. The idea underlying these models is that user preferences are determined by a latent user vector and a latent item vector in a low dimensional space that captures the intrinsic structure of users' interests. [13] introduced the probabilistic matrix factorization (PMF) model which scales linearly and performs well on large and sparse data sets. Bayesian approaches address the weakness of requiring much parameter tuning in standard matrix factorization. [12] presented Bayesian probabilistic matrix factorization (BPMF) model, extending PMF to a fully Bayesian treatment that achieves significantly higher prediction accuracy than PMF. Another focus in the literature is improving recommendation accuracy by incorporating additional information, such as side information [11], content information of items [17], and social networks [4]. However, all the above methods only learn a global model from a series of unordered ratings without considering the evolution of users' preferences over time.

[2] uses a time weighting scheme for a similarity based collaborative filtering approach, which decays the similarities to previously rated items as time difference increases at the prediction time. As discussed above, the time decay scheme may miss a long-term effect for some users. Our method learns the temporal dependence from the whole set of ratings without limiting any part of the data. [19] proposed the user-item-time tensor factorization to model temporal effects. In a recommender system, the time dimension is a local effect and should not be compared across all (user,item) pairs [18]. [18] used a graph connecting users, items, and sessions to model users' long-term preferences and short-term preferences. [15] modeled temporal effects using Kalman filtering with a transition process parameter for each user similar to our transition matrix, but their transition parameters are time-dependent and user-supplied, and their model was evaluated only on generated data. Clearly, specifying such parameters for all users at all time points is impractical. In contrast, our transition matrices are time-invariant and are learnt automatically by the model from observed data. [1] further extended [15] in the scenario of temporal adoption.

Our work is most related to [6], which presented the first temporal model for the matrix factorization approach by introducing time-variant biases for each user and each item. The time-variant biases are the differences at each time window, but do not summarize the underlying pattern for such differences. This approach works for prediction only if two windows share similar biases. In contrast, our time-invariant transition matrices capture properties that are independent of time, and thus, can be used for prediction in a new time window. Beyond prediction, time-invariant properties also helps understand the mechanism that underpins the temporal dynamics. More discussions on this point will be given in Section 2.1.

The rest of the paper is organized as follows. Section 2 introduces the temporal probabilistic matrix factorization. Section 3 extends the temporal probabilistic matrix factorization to its fully Bayesian model. Section 4 presents our experimental studies. Finally, we conclude the paper.

2 Temporal Probabilistic Matrix Factorization

We present the temporal probabilistic matrix factorization (TMF) to model user temporal dynamics. We assume that time is represented by a series of consecutive time windows. Matrix factorization models map both users and items to a joint latent factor space of a low dimensionality D, such that ratings are modeled as inner products in that space. Suppose we have M items, N users, S time windows, and rating values from 1 to K.

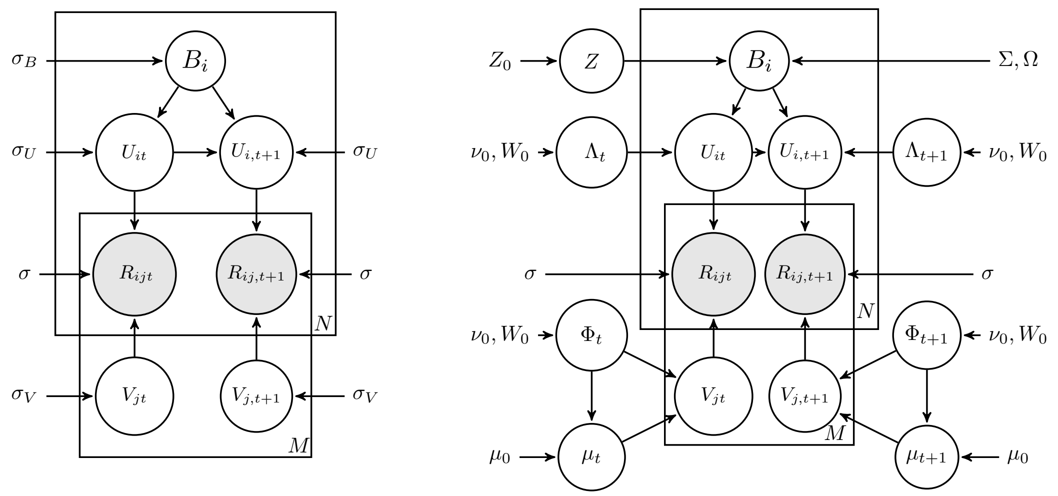


Figure 2: Graphical representations of TMF (left) and BTMF (right), with parameters and hyperparameters of time window t and $t+1$ shown only

Let R_{ijt} denote the rating of user i for item j at time window t, $U \in \mathbb{R}^{D\times N\times S}$ and $V \in \mathbb{R}^{D\times M\times S}$ denote latent user and latent item (factor) hypermatrices, with column vectors U_{it} and V_{jt} representing user-specific and item-specific latent (factor) vectors, which describe user's interests and item's features, at time window t, respectively.

2.1 Introducing Transition Matrix We assume that there is a temporal dependence between the latent user vectors U_{it} and $U_{i,t-1}$ and we model this dependency by a transition hypermatrix $B \in \mathbb{R}^{N\times D\times D}$ for all users. In particular, B_i, the $D \times D$ transition matrix for user i, models the transition of user i's preferences in the *latent* space from the previous time window to the next; so we expect U_{it} to have the mean $B_i U_{i,t-1}$. In plain English, this says that for each user i, the jth latent factor in the next time window is a linear combination, as specified by the jth row of B_i, of the latent factors in the previous time window. As each latent factor captures some intrinsic feature of items (e.g., movie's genre, movie's director, etc.), B_i captures the time-invariant aspect of the user i's evolution in this intrinsic feature space. It makes sense not to model this dependency for items since item's features are stable and less correlated.

For example, consider the $D = 2$ latent space. The identity transition matrix $\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$ represents a stable latent user vector that does not change much over time; the transition matrix $\begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$ represents the alternating changing pattern between two latent user vectors (a, b) and (b, a); the transition matrix $\begin{bmatrix}1.1 & 0\\0 & 1\end{bmatrix}$ represents a gradual shift pattern toward the first factor. In a higher dimensional latent space, a different subspace could undergo a different pattern, therefore, several patterns could occur at the same time.

At the current time t, we learn the latent user vector U_{it} and latent item vector V_{jt} for each user i and each item j, as well as the transition matrix B_i from *all* the ratings collected up to time t. With the learnt parameters, we can predict the rating of the user i on the item j at a future time window by the rule $R^*_{ij} = (B_i U_{it})^T V_{jt}$. The model performance is measured by the *root mean squared error (RMSE)* on the testing set $\{R_{ij}\}$:

$$RMSE = \sqrt{\frac{\sum_{i,j}(R_{ij} - R^*_{ij})^2}{n}} \tag{2.1}$$

where n is the number of ratings in the testing set. Using all the ratings collected up to the current time t helps capture the long-term effect discussed in Section 1 while modeling temporal changes through transition matrices. As the current time advances to the time $t+1$, the above learning process is repeated on all the ratings collected up to time $t+1$. The choice of the granularity of time dictates the trade-off between the freshness of updates and the efficiency of learning.

2.2 Modeling We develop a temporal probabilistic model with Gaussian observation noises to learn the parameters U, V, and B. This is done by extending the conventional probabilistic matrix factorization (PMF) model [13] with the transition hypermatrix B. The conditional distribution over the observed ratings R is

$$p(R|U, V, \sigma) = \prod_{t=1}^{S}\prod_{i=1}^{N}\prod_{j=1}^{M}[\mathcal{N}(R_{ijt}|U_{it}^T V_{jt}, \sigma^2)]^{I_{ijt}} \tag{2.2}$$

where $\mathcal{N}(x|\mu, \sigma^2)$ is the probability density function of the Gaussian distribution with mean μ and variance σ^2, and I_{ijt} is the indicator variable that is equal to 1 if user i rated item j at time window t and equal to 0 otherwise.

As in [13], we place zero-mean spherical Gaussian priors on latent item vectors in V:

$$p(V|\sigma_V) = \prod_{t=1}^{S}\prod_{j=1}^{M} p(V_{jt}|\sigma_V) = \prod_{t=1}^{S}\prod_{j=1}^{M} \mathcal{N}(V_{jt}|\mathbf{0}, \sigma_V^2\mathbf{I}) \tag{2.3}$$

To model latent user vectors in U, we place Gaussian priors with mean $B_iU_{i,t-1}$ on latent user vectors U_{it} to model the temporal dependence of time window t on time window $t-1$:

$$p(U_{it}|B_i, U_{i,t-1}, \sigma_U) = \mathcal{N}(U_{it}|B_iU_{i,t-1}, \sigma_U^2\mathbf{I}) \tag{2.4}$$

where $p(U_{i1}|B_i, U_{i0}, \sigma_U) = \mathcal{N}(U_{i1}|\mathbf{0}, \sigma_U^2\mathbf{I})$ by defining $U_{i0} = \mathbf{0}$. Integrating out variables i and t gives:

$$\begin{aligned} p(U|B, \sigma_U) &= \prod_{t=1}^{S}\prod_{i=1}^{N} p(U_{it}|B_i, U_{i,t-1}, \sigma_U) \\ &= \prod_{t=1}^{S}\prod_{i=1}^{N} \mathcal{N}(U_{it}|B_iU_{i,t-1}, \sigma_U^2\mathbf{I}) \end{aligned} \tag{2.5}$$

The latent transition matrix B_i is placed with matrix-variate normal distribution $\mathcal{MN}(Z, \Sigma, \Omega)$ where Z is a matrix containing the expectation of each element of B_i, and Σ, Ω are two covariance matrices. In this case, we set $Z = \mathbf{I}$, i.e., a $D \times D$ identity matrix , and $\Sigma = \Omega = \sigma_B\mathbf{I}$:

$$p(B|\sigma_B) = \prod_{i=1}^{N} \mathcal{MN}(B_i|\mathbf{I}, \sigma_B\mathbf{I}, \sigma_B\mathbf{I}) \tag{2.6}$$

2.3 Inference Following Eq. (2.2-2.6) and the graphical representation of TMF shown in Fig. 2 , the posterior distribution over the user and item vectors is given by

$$\begin{aligned} &p(U, V, B|R, \sigma, \sigma_V, \sigma_U, \sigma_B) \\ &\propto p(U|B, \sigma_U)p(V|\sigma_V)p(B|\sigma_B)p(R|U, V, \sigma) \end{aligned} \tag{2.7}$$

Our goal is to find the values of U_{it}, V_{jt}, and B_i that maximize the log-posterior of Eq. (2.7), which is equivalent to minimizing the sum-of-squared-errors objective function with quadratic regularization terms:

$$\begin{aligned} &\frac{1}{2}\sum_{t=1}^{S}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ijt}(R_{ijt} - U_{it}^TV_{jt})^2 + \frac{\lambda_V}{2}\sum_{t=1}^{S}\sum_{j=1}^{M}\|V_{jt}\|_{Fro}^2 \\ &+ \frac{\lambda_U}{2}\sum_{t=1}^{S}\sum_{i=1}^{N}\|U_{it} - B_iU_{i,t-1}\|_{Fro}^2 + \frac{\lambda_B}{2}\sum_{i=1}^{N}\|B_i - \mathbf{I}\|_{Fro}^2 \end{aligned} \tag{2.8}$$

where $\lambda_U = \sigma^2/\sigma_U^2$, $\lambda_V = \sigma^2/\sigma_V^2$, $\lambda_B = \sigma^2/\sigma_B^2$ and $\|\cdot\|_{Fro}^2$ denotes the Frobenius norm. We adopt gradient descent with learning rate η in U, V and B to find the local minimum of the objective function in Eq. (2.8).

3 The Fully Bayesian Model (BTMF)

One drawback of TMF is that it is hard to search appropriate values of the hyperparameters $\sigma, \sigma_U, \sigma_V, \sigma_B$ to control the model complexity. A possible solution is to integrates out all model parameters U, V, B and hyperparameters $\sigma, \sigma_U, \sigma_V, \sigma_B$ to achieve the *predictive distribution* given observed data. In this section, we extend TMF to a fully Bayesian treatment called BTMF, in which both parameters and hyperparameters are sampled from the predictive distribution through the MCMC method. Though the mathematical development is a bit involved, the spirit of the extension is essentially the same as extending the probabilistic matrix factorization to its fully Bayesian treatment [12].

3.1 Modeling BTMF introduces priors for the hyperparameters to control the model complexity, as shown in Fig. 2. Instead of Eq. (2.3) with fixed settings, BTMF models two hyperparameters, the mean vector μ_t and the precision matrix Φ_t, for each latent item vector V_{jt}, as in [12]; the prior distribution is assumed to be Gaussian:

$$p(V_{jt}|\mu_t, \Phi_t) = \mathcal{N}(V_{jt}|\mu_t, \Phi_t^{-1}) \tag{3.9}$$

And we place Gaussian-Wishart priors on μ_t, Φ_t:

$$\begin{aligned} &p(\mu_t, \Phi_t|\mu_0, \beta_0, W_0, \nu_0) \\ &= p(\mu_t|\Phi_t, \mu_0, \beta_0)p(\Phi_t|W_0, \nu_0) \\ &= \mathcal{N}(\mu_t|\mu_0, (\beta_0\Phi_t)^{-1})\mathcal{W}(\Phi_t|W_0, \nu_0) \end{aligned} \tag{3.10}$$

Here $\mathcal{W}$ is the Wishart distribution with ν_0 degrees of freedom and a $D \times D$ scale matrix W_0:

$$\mathcal{W}(\Lambda|W_0, \nu_0) = \frac{1}{C}|\Lambda|^{(\nu_0-D-1)/2}\exp(-\frac{1}{2}\mathrm{Tr}(W_0^{-1}\Lambda))$$

For each latent user vector U_{it}, the mean vector is given by $B_iU_{i,t-1}$ and we place Wishart priors on the user hyperparameter Λ_t:

$$p(U_{it}|B_i, U_{i,t-1}, \Lambda_t) = \mathcal{N}(U_{it}|B_iU_{i,t-1}, \Lambda_t^{-1}) \tag{3.11}$$

$$p(\Lambda_t|W_0, \nu_0) = p(\Lambda_t|W_0, \nu_0) = \mathcal{W}(\Lambda_t|W_0, \nu_0) \tag{3.12}$$

For each transition matrix B_i, there are three hyperparameters, i.e., the mean matrix Z and two covariance matrices Σ, Ω; contrary to Eq. (2.6), the prior distribution is assumed to be matrix normal:

$$p(B_i|Z, \Sigma, \Omega) = \mathcal{MN}(B_i|Z, \Sigma, \Omega) \tag{3.13}$$

To be simplified, we place no priors on Σ and Ω (Indeed, the priors for variance matrices of matrix normal distribution are hyper inverse Wishart distributions but they have little effects). We place the prior Z_0 to control the expectation matrix Z and set $\Sigma = \Omega = \mathbf{I}$.

For the sake of convenience, we define the hyperparameters $\Theta_U = \{\Lambda_{t=1...S}\}$, $\Theta_V = \{\mu_{t=1...S}, \Phi_{t=1...S}\}$ and $\Theta_B = \{Z\}$ controlled by priors $\Theta_0 = \{\mu_0, \nu_0, \beta_0, W_0, Z_0\}$. The predictive distribution of the rating value R^*_{ij} for user i and item j at future time window can be obtained by marginalization:

(3.14)
$$p(R^*_{ij}|R, \Theta_0) = \iint p(R^*_{ij}|U_{it}, V_{jt}, B_i) p(U, V, B|R, \Theta_U, \Theta_V, \Theta_B) p(\Theta_U, \Theta_V, \Theta_B|\Theta_0) \mathrm{d}\{U, V, B\} \mathrm{d}\{\Theta_U, \Theta_V, \Theta_B\}$$

The exact evaluation of this predictive distribution is analytically intractable due to the complexity of the posterior. MCMC-based methods [10] use the Monte Carlo approximation to the predictive distribution given by

$$p(R^*_{ij}|R, \Theta_0) \approx \frac{1}{\vartheta} \sum_{\kappa=1}^{\vartheta} p(R^*_{ij} \mid U^\kappa_{it}, V^\kappa_{jt}, B^\kappa_i) \quad (3.15)$$

where ϑ is the given maximal iteration number and $U^\kappa_{it}, V^\kappa_{jt}, B^\kappa_i$ are samples at κth iteration.

3.2 Inference To compute R^*_{ij} using Eq. (3.15), we need to sample the variables U, V, B and Θ_U, Θ_V, Θ_B in turn from its distribution conditional on the current values of all other variables, according to Gibbs sampling. Below, we describe these conditional distributions.

Sampling U_{it} and hyperparameter Λ_t: Due to the use of conjugate priors for the parameters and hyperparameters in our model, the conditional distribution over the latent user vector U_{it}, conditioned on other variables $(V, R, B...)$ and the hyperparameters (Θ_U, σ), is Gaussian:

(3.16)
$$p(U_{it}|V, R, B, U_{i,t-1}, \Theta_U, \sigma) = \mathcal{N}(U_{it}|\mu^*_U, [\Lambda^*_U]^{-1}) \propto \prod_{j=1}^{M} [\mathcal{N}(R_{ijt}|U^T_{it} V_{jt}, \sigma^2)]^{I_{ijt}} p(U_{it}|B_i, U_{i,t-1}, \Lambda_t)$$

where
$$\Lambda^*_U = \Lambda_t + \frac{1}{\sigma^2} \sum_{j=1}^{M} I_{ijt}[V_{jt} V^T_{jt}],$$
$$\mu^*_U = [\Lambda^*_U]^{-1} (\frac{1}{\sigma^2} \sum_{j=1}^{M} I_{ijt}[V_{jt} R_{ijt}] + \Lambda_t B_i U_{i,t-1}).$$

The conditional distribution over the user hyperparameters Λ_t conditioned on the latent user feature hypermatrix U and transition hypermatrix B is given by the Wishart distribution:

$$p(\Lambda_t|U, B, \Theta_0) = \mathcal{W}(\Lambda_t|W^*_0, \nu^*_0) \quad (3.17)$$

where
$$[W^*_0]^{-1} = W^{-1}_0 + \sum_{i=1}^{N} (U_{it} - B_i U_{i,t-1})(U_{it} - B_i U_{i,t-1})^T,$$
$$\nu^*_0 = \nu_0 + N.$$

Sampling V_{jt} and hyperparameters μ_t, Φ_t: This part is the same as the conventional fully Bayesian case [12]. The conditional distribution over the latent item vector V_{jt}, conditioned on other variables (U, R) and the hyperparameters (Θ_V, σ), is Gaussian:

$$p(V_{jt}|U, R, \Theta_V, \sigma) = \mathcal{N}(V_{jt}|\mu^*_V, [\Phi^*_V]^{-1}) \propto \prod_{i=1}^{N} [\mathcal{N}(R_{ijt}|U^T_{it} V_{jt}, \sigma^2)]^{I_{ijt}} p(V_{jt}|\mu_t, \Phi_t) \quad (3.18)$$

where
$$\Phi^*_V = \Phi_t + \frac{1}{\sigma^2} \sum_{i=1}^{N} I_{ijt}[U_{it} U^T_{it}],$$
$$\mu^*_V = [\Phi^*_V]^{-1} (\frac{1}{\sigma^2} \sum_{i=1}^{N} I_{ijt}[U_{it} R_{ijt}] + \Phi_t \mu_t).$$

The conditional distribution over the item hyperparameters μ_t, Φ_t conditioned on the latent item vector hypermatrix V is given by the Gaussian-Wishart distribution:

(3.19)
$$p(\mu_t, \Phi_t|V, \Theta_0) = \mathcal{N}(\mu_t|\mu^*_0, (\beta^*_0 \Phi_t)^{-1}) \mathcal{W}(\Phi_t|W^*_0, \nu^*_0)$$

where
$$\mu^*_0 = \frac{\beta_0 \mu_0 + M\bar{V}}{\beta_0 + M}, \quad \beta^*_0 = \beta_0 + M, \quad \nu^*_0 = \nu_0 + M,$$
$$[W^*_0]^{-1} = W^{-1}_0 + \bar{C} + \frac{\beta_0 M}{\beta_0 + M} (\mu_0 - \bar{V})(\mu_0 - \bar{V})^T,$$
$$\bar{V} = \frac{1}{M} \sum_{j=1}^{M} V_{jt}, \quad \bar{C} = \sum_{j=1}^{M} (V_{jt} - \bar{V})(V_{jt} - \bar{V})^T.$$

Sampling B_i and hyperparameter Z: Like Eq. (3.16) and (3.18), we assume that the conditional distribution over each transition matrix B_i is matrix normal, and the conditional distribution is determined by a series of multivariate normal distribution and a prior distribution according to $p(B_i|U, \Theta_B) \propto p(U|B_i) p(B_i|\Theta_B)$. The approximation to this conditional distribution is as follows:

$$p(B_i|U, \Theta_B) = \mathcal{MN}(B_i|Z^*, \Sigma, \Omega) \propto \prod_{t=1}^{S} \mathcal{N}(U_{it}|B_i U_{i,t-1}, \Lambda^{-1}_U) p(B_i|Z, \Sigma, \Omega) \quad (3.20)$$

where

$$Z^* = (\sum_{t=1}^{S} [U_{it} U^T_{i,t-1}] + Z)(\sum_{t=1}^{S} [U_{i,t-1} U^T_{i,t-1}] + \mathbf{I})^{-1},$$
$$\Sigma = \Omega = \mathbf{I}.$$

Analogously, the conditional distribution over Z conditioned on the transition hypermatrix B is given by the matrix normal distribution:

$$p(Z|B, \Theta_0) = \mathcal{MN}(Z|Z_0^*, \Sigma, \Omega) \tag{3.21}$$

where

$$Z_0^* = \frac{Z_0 + N\bar{B}}{1+N}, \quad \bar{B} = \frac{1}{N}\sum_{i=1}^{N} B_i, \quad \Sigma = \Omega = \mathbf{I}.$$

Algorithm 1 shows the process of the Gibbs sampling for BTMF. $x \sim y$ means sampling the random variable x following the distribution y.

Algorithm 1 Gibbs sampling for BTMF

```
Initialize the model parameters { $U^1, V^1, B^1$ }
for $\kappa = 1 \to \vartheta$ do
    Sample the hyperparameters by Eq. (3.17) (3.19) (3.21) for all time windows:
        $\Theta_U^\kappa \sim p(\Theta_U|U^\kappa, B^\kappa, \Theta_0)$,
        $\Theta_V^\kappa \sim p(\Theta_V|V^\kappa, \Theta_0)$,
        $\Theta_B^\kappa \sim p(\Theta_B|B^\kappa, \Theta_0)$.
    For each time window $t = 1, ..., S$, sample each latent user vector by Eq. (3.16), sample each latent item vector by Eq. (3.18):
        $U_{it}^{\kappa+1} \sim p(U_{it}|V^\kappa, R, B^\kappa, U_{i,t-1}^\kappa, \Theta_U^\kappa, \sigma)$,
        $V_{jt}^{\kappa+1} \sim p(V_{jt}|U^\kappa, R, \Theta_V^\kappa, \sigma)$.
    For each user $i = 1, ..., N$, sample the transition matrix in parallel by Eq. (3.20):
        $B_i^{\kappa+1} \sim p(B_i|U^\kappa, \Theta_B^\kappa)$.
end for
```

4 Experiments

We conducted extensive experiments to evaluate the prediction accuracy of the proposed methods. We first introduce the experimental settings and then present the main findings.

4.1 Experimental Setup

We evaluate **TMF** proposed in Section 2 and **BTMF** proposed in Section 3 against the following methods: **PMF** refers to the probabilistic matrix factorization model widely used in collaborative filtering [13]. **BPMF** refers to the fully Bayesian treatment to PMF achieving better results [12]. **timeSVD** refers to the time sensitive algorithm applied successfully in Netflix data set [6]. **Ensemble** refers to an ensemble method by adopting PMF in separate windows and tuning the relevant weights.

We conduct experiments on six data sets: MovieLens[1] (movie ratings), BeerAdvocate[2] (beer ratings), FineFoods[2] (food ratings), Epinions [4] (product ratings), EachMovie[1] (movie ratings), and Flixster [4] (movie ratings). We keep the first two data sets unchanged because they have a balanced scale, and preprocess the other four data sets by removing the users with less than 20 ratings as in [8]. All data sets contain ratings ranging from 1 to 5, except for the Eachmovie data set which ranges from 1 to 6. The statistics of the processed data sets are shown in Table 1.

Table 1: Statistics of data sets.

Data set	#User	#Item	#Rating	Timespan
MovieLens	2113	9801	824600	1997.11-2008.12
BeerAdvocate	33387	66051	1586259	1998.1-2012.10
FineFoods	7590	27385	141294	1999.10-2012.10
Epinions	14077	96291	470557	1999.3-2000.12
EachMovie	36658	1623	2580267	1996.2-1997.8
Flixster	36492	48277	7729741	2005.12-2009.11

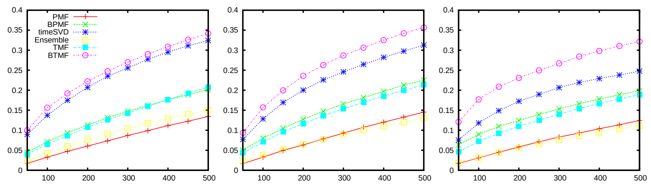
The first three data sets span a longer period of time than the last three. For each data set, we created $S = 10$ time windows as follows. We partition each of the first three data sets yearly and merge several oldest windows (which have less data) into one window to create a total of 10 time windows. For the last three data sets, we partition them bimonthly or half-yearly to get 10 time windows. The last three windows are used for testing. For each testing window, we use all the time windows prior to it, except for Ensemble, as the training set, and run the experiment five times and report the average results for reliability. For Ensemble, we use each of the q most recent windows prior to the testing window to learn a model and average the scores of these models. We report the best result for all possible q.

We set $\eta = 0.001, \lambda_U = \lambda_V = 0.01$ in PMF, timeSVD, Ensemble and TMF, with other internal parameters with default settings in timeSVD and $\lambda_B = 0.01$ in TMF, and set $\nu_0 = D, \beta_0 = 2, W_0 = \mathbf{I}, \mu_0 = \mathbf{0}$ in BPMF and BTMF, with additional $Z_0 = \mathbf{I}$ in BTMF.

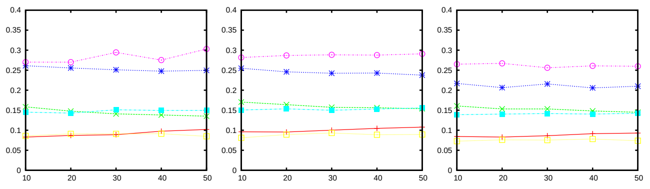
We evaluate the accuracy of estimated ratings based on RMSE and Recall@k for the testing data. *RMSE*, defined in Eq. (2.1), is the root mean squared error measuring the difference between the estimated rating values and the true rating values, thus, the prediction accuracy on the individual rating level. *Recall@k* for a user u is defined as the ratio $\frac{|N(k;u)|}{|N(u)|}$, where $|\cdot|$ denotes the number of elements in a set, $N(u)$ is the set of items rated by u in the testing set and $N(k;u)$ is the subset of $N(u)$ contained in the top-k list of all items sorted by their estimated ratings. We report the average of the recall over all users in the testing set. In practice, recall is more useful in recommender systems since it takes a global view on all items and a high recall indeed reflects

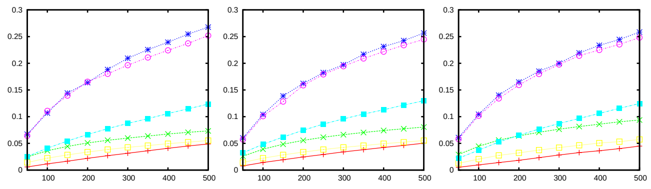
[1]http://www.grouplens.org/node/12

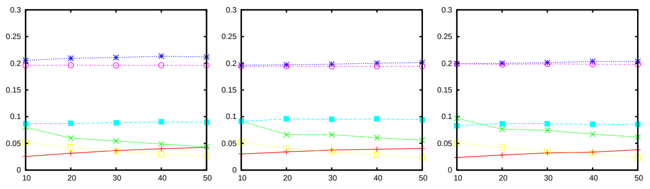
[2]http://snap.stanford.edu/data/#reviews [9]

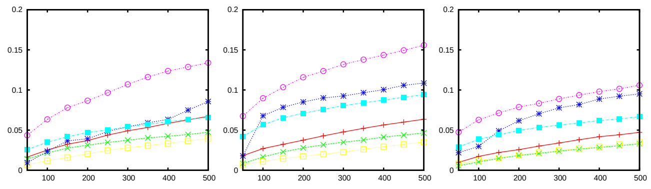


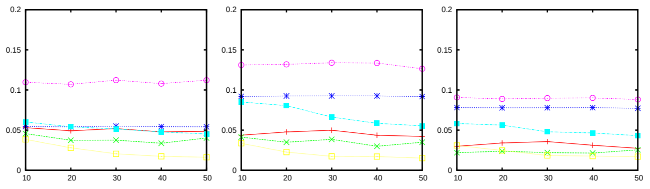
(a) Recall@k at MovieLens, $D = 20$. Vary k

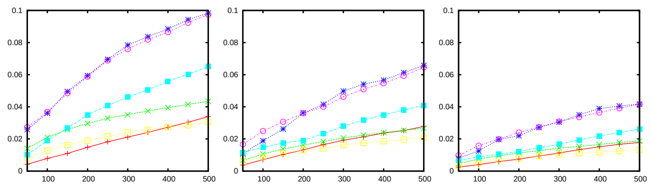

(b) Recall@300 at MovieLens. Vary D

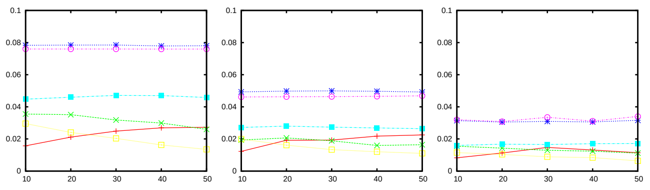

(c) Recall@k at BeerAdvocate, $D = 20$. Vary k

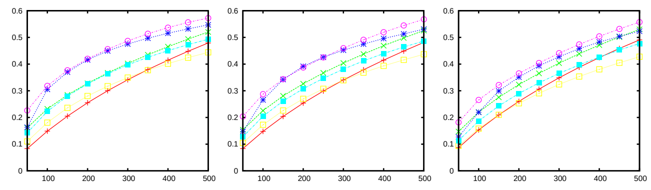

(d) Recall@300 at BeerAdvocate. Vary D

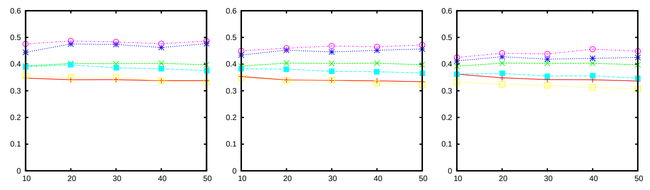

(e) Recall@k at FineFoods, $D = 20$. Vary k

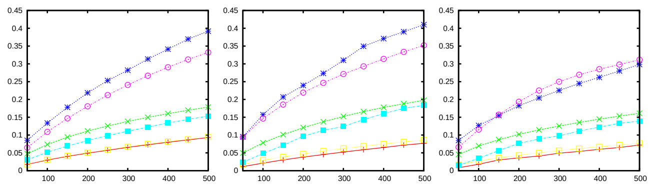

(f) Recall@300 at FineFoods. Vary D


(g) Recall@k at Epinions, $D = 20$. Vary k


(h) Recall@300 at Epinions. Vary D


(i) Recall@k at Eachmovie, $D = 20$. Vary k


(j) Recall@300 at Eachmovie. Vary D


(k) Recall@k at Flixster, $D = 20$. Vary k

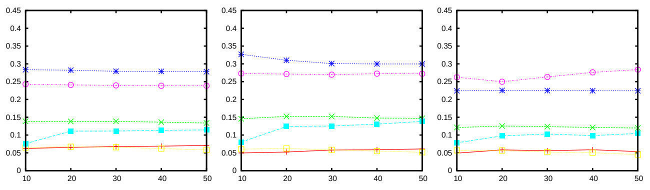

(l) Recall@300 at Flixster. Vary D

Figure 3: Recall of different models. Each row refers to a data set and each column refers to a testing window. The left part observes the effect of varying k with fixed $D = 20$ while the right part observes the effect of varying D with fixed $k = 300$. Higher values are better.

Table 2: RMSE (mean$\pm$standard error) of different models. $D = 20$. The best performer is in **boldface** and the second best performer is in *italic*.

	MovieLens			**BeerAdvocate**		
Algorithm	**Window 1**	**Window 2**	**Window 3**	**Window 1**	**Window 2**	**Window 3**
PMF[13]	0.9661 ± 0.0018	0.9478 ± 0.0026	0.9036 ± 0.0016	0.7006 ± 0.0015	0.6247 ± 0.0014	0.6368 ± 0.0010
BPMF[12]	0.9351 ± 0.0029	0.9350 ± 0.0057	0.8843 ± 0.0015	0.6917 ± 0.0005	0.6226 ± 0.0005	0.6325 ± 0.0010
timeSVD[6]	0.9170 ± 0.0021	0.9049 ± 0.0010	0.8683 ± 0.0013	$\mathbf{0.6424} \pm 0.0013$	$\mathbf{0.5634} \pm 0.0017$	$\mathbf{0.5648} \pm 0.0009$
Ensemble	0.9531 ± 0.0007	0.9642 ± 0.0020	0.9105 ± 0.0027	0.6984 ± 0.0007	0.6269 ± 0.0008	0.5941 ± 0.0032
TMF	$\mathit{0.8909} \pm 0.0020$	$\mathit{0.8801} \pm 0.0020$	$\mathit{0.8557} \pm 0.0027$	0.6660 ± 0.0007	0.5800 ± 0.0012	0.5882 ± 0.0022
BTMF	$\mathbf{0.8712} \pm 0.0006$	$\mathbf{0.8454} \pm 0.0006$	$\mathbf{0.8310} \pm 0.0004$	$\mathit{0.6524} \pm 0.0003$	$\mathit{0.5708} \pm 0.0003$	$\mathit{0.5779} \pm 0.0002$
	FineFoods			**Epinions**		
PMF[13]	1.2671 ± 0.0032	1.2078 ± 0.0073	1.2429 ± 0.0097	1.2419 ± 0.0087	1.2268 ± 0.0037	1.1924 ± 0.0048
BPMF[12]	1.2537 ± 0.0003	1.1792 ± 0.0002	1.1940 ± 0.0006	1.1892 ± 0.0004	1.1818 ± 0.0005	1.1542 ± 0.0003
timeSVD[6]	$\mathit{1.2484} \pm 0.0008$	$\mathbf{1.1684} \pm 0.0009$	$\mathit{1.1893} \pm 0.0007$	1.1707 ± 0.0013	1.1655 ± 0.0017	1.1357 ± 0.0009
Ensemble	1.2528 ± 0.0004	1.1779 ± 0.0018	1.1956 ± 0.0020	1.1929 ± 0.0017	1.1847 ± 0.0008	1.1582 ± 0.0002
TMF	1.2506 ± 0.0047	1.1820 ± 0.0058	1.2089 ± 0.0041	$\mathit{1.1125} \pm 0.0016$	$\mathit{1.1117} \pm 0.0012$	$\mathit{1.0978} \pm 0.0015$
BTMF	$\mathbf{1.2476} \pm 0.0004$	$\mathit{1.1744} \pm 0.0002$	$\mathbf{1.1891} \pm 0.0001$	$\mathbf{1.1024} \pm 0.0002$	$\mathbf{1.1067} \pm 0.0002$	$\mathbf{1.0937} \pm 0.0001$
	EachMovie			**Flixster**		
PMF[13]	1.3163 ± 0.0015	1.3099 ± 0.0044	1.3608 ± 0.0025	1.1299 ± 0.0082	1.0511 ± 0.0056	1.0899 ± 0.0067
BPMF[12]	$\mathit{1.2894} \pm 0.0016$	$\mathbf{1.2855} \pm 0.0017$	$\mathbf{1.3134} \pm 0.0014$	1.1100 ± 0.0008	$\mathit{1.0337} \pm 0.0015$	$\mathit{1.0489} \pm 0.0024$
timeSVD[6]	1.3696 ± 0.0010	1.3736 ± 0.0026	1.4233 ± 0.0019	$\mathit{1.0865} \pm 0.0007$	1.0390 ± 0.0003	1.0539 ± 0.0006
Ensemble	1.4659 ± 0.0053	1.4778 ± 0.0064	1.5371 ± 0.0052	1.1466 ± 0.0024	1.0572 ± 0.0022	1.0713 ± 0.0006
TMF	1.3477 ± 0.0024	1.3493 ± 0.0019	1.3909 ± 0.0030	1.1555 ± 0.0038	1.0723 ± 0.0011	1.1290 ± 0.0004
BTMF	$\mathbf{1.2867} \pm 0.0080$	$\mathit{1.2892} \pm 0.0098$	$\mathit{1.3414} \pm 0.0143$	$\mathbf{1.0802} \pm 0.0038$	$\mathbf{1.0329} \pm 0.0022$	$\mathbf{1.0436} \pm 0.0018$

the user's adoption.

4.2 Experimental Results The followings are the findings on Recall@k and RMSE.

Recall@k. Fig. 3 presents the recall performance of different models at three testing windows of six data sets. The left three columns show Recall@k at the latent dimensionality $D = 20$ while varying k from 50 to 500. The right three columns show Recall@300 while varying D from 10 to 50. We observed that our proposed model TMF always performs better than its conventional counterpart PMF that has no temporal consideration. Ensemble, whose data partitioning misses global patterns and suffers from the data sparsity issue, works no better than PMF. These evidences suggest that considering the effect of temporal dependence and dynamics through the transition matrix help improve the accuracy of recommendation.

The fully Bayesian treatment BTMF beats PMF, BPMF, Ensemble, and TMF in all six data sets. The performance of BTMF is steady with respect to the dimensionality D. The relative ranks of items are better predicted by our temporal models due to the reality that user's subsequent ratings depend more on recent ratings, which is modeled by $R^*_{ij} = (B_i U_{it})^T V_{jt}$, where t is the current time. However, simply focusing on recent ratings, like Ensemble, will miss patterns represented by anterior ratings. Our models address this problem by capturing the temporal dependence and the users' interest shift through the transition matrix B_i learnt from the full set of rating data.

The results of BTMF are competitive even compared with the time-aware timeSVD: BTMF beats timeSVD on MovieLens, FineFood and EachMovie data sets and performs similarly on other data sets. timeSVD aims to capture temporal effects by incorporating a time-variant bias for each user and item at every individual time window, whereas BTMF captures temporal effects through incorporating a single time-invariant transition matrix for each user. Unlike time-variant biases, the time-invariant transition matrix captures the temporal pattern that holds all times, and thus, is suitable for prediction in the next time window. In contrast, time-variant biases are differences at each individual time window and do not summarize the underlying pattern for such differences. This approach works for prediction only if two windows share similar biases. Indeed, for the faster changing FineFoods and MovieLens, we observed a more significant improvement of BTMF over timeSVD.

RMSE. Table 2 shows the RMSE of different models. As pointed out in [6], achievable RMSE values lie in a quite compressed range and small improvements in RMSE terms can have a significant impact on the quality of the top few presented recommendations. First of all, the two fully Bayesian matrix factorization models, BPMF and BTMF, achieve better performance than their non-Bayesian counterparts, PMF and TMF.

The boldface and italic highlight the best and second best performers, respectively. Since timeSVD and BTMF perform best on recall, we focus on these two methods here. With 6 data sets and 3 testing windows for each, there are 18 testing cases. Among these 18 testing cases, BTMF performs best in 12 cases and second best in 6 cases, whereas timeSVD performs best in 4 cases and second best in 3 cases. For the four cases where timeSVD performs best (i.e., three cases for BeerAdvocate and one case for FineFoods), timeSVD is only slightly better than BTMF.

In summary, the proposed BTMF outperforms previous models in most cases. Two elements contribute to this improvement. One is considering temporal effects in the full time range without discarding any part of the data. Ensemble, which uses several recent time windows, performs poorly. The second element is the time-invariant transition matrix that captures a property essential for prediction. The time-variant biases used by the timeSVD model are fit for prediction only if such biases are similar for adjacent time windows. The transition matrix also provides a way to understand the pattern of evolution for a user. We observed two types of transition matrices learnt. The first type has non-zero entries on the main diagonal, which captures those inactive users who have very few ratings observed. The second type has non-zero entries outside the main diagonal and such entries represent certain preference shifts among latent factors, and a different distribution of non-zero entries captures a different shift pattern.

5 Conclusion

We proposed two temporal matrix factorization methods, TMF and BTMF, to predict user preferences that evolve over time. The key idea is to model the evolution by a latent transition matrix that captures the time-invariant property of user's temporal dynamics, thus, the "pattern of evolution" for a user. We presented inference algorithms for these methods and evaluated their effectiveness. The experimental results demonstrated improved prediction over previous methods.

Acknowledgements

This work is partially supported by a Discovery Grant from Natural Sciences and Engineering Research Council of Canada, partially supported by the Singapore National Research Foundation under its International Research Centre@ Singapore Funding Initiative and administered by the IDM Programme Office, Media Development Authority (MDA). The first author thanks Zhao Song for model discussion in the early stage.

References

[1] F. C. T. Chua, R. Oentaryo, and E.-P. Lim. Modeling temporal adoptions using dynamic matrix factorization. In *ICDM*, 2013.

[2] Y. Ding and X. Li. Time weight collaborative filtering. In *CIKM*, pages 485–492, 2005.

[3] Z. Ghahramani and M. I. Jordan. Factorial hidden markov models. *Machine learning*, 29(2-3):245–273, 1997.

[4] M. Jamali and M. Ester. A transitivity aware matrix factorization model for recommendation in social networks. In *IJCAI*, pages 2644–2649, 2011.

[5] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[6] Y. Koren. Collaborative filtering with temporal dynamics. In *KDD*, pages 447–456, 2009.

[7] B. Marlin. Modeling user rating profiles for collaborative filtering. In *NIPS*, volume 16, 2003.

[8] B. Marlin and R. S. Zemel. The multiple multiplicative factor model for collaborative filtering. In *ICML*, pages 73–80, 2004.

[9] J. McAuley and J. Leskovec. From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews. *In WWW*, 2013.

[10] R. M. Neal. Probabilistic inference using markov chain monte carlo methods. *Technical Report CRG-TR-93-1*, 1993.

[11] I. Porteous, A. Asuncion, and M. Welling. Bayesian matrix factorization with side information and dirichlet process mixtures. In *AAAI*, pages 563–568, 2010.

[12] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *ICML*, pages 880–887, 2008.

[13] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. *NIPS*, pages 1257–1264, 2008.

[14] N. Srebro, J. D. Rennie, and T. Jaakkola. Maximum-margin matrix factorization. *NIPS*, 17(5):1329–1336, 2005.

[15] J. Z. Sun, K. R. Varshney, and K. Subbian. Dynamic matrix factorization: A state space approach. In *ICASSP*, pages 1897–1900, 2012.

[16] A. Tsymbal. The problem of concept drift: definitions and related work. *Technical Report TCD-CS-2004-15, Trinity College Dublin*, 2004.

[17] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *KDD*, pages 448–456, 2011.

[18] L. Xiang, Q. Yuan, S. Zhao, L. Chen, X. Zhang, Q. Yang, and J. Sun. Temporal recommendation on graphs via long-and short-term preference fusion. In *KDD*, pages 723–732, 2010.

[19] L. Xiong, X. Chen, T.-K. Huang, J. G. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM*, volume 10, pages 211–222, 2010.

[20] M. Xu, J. Zhu, and B. Zhang. Nonparametric max-margin matrix factorization for collaborative prediction. In *NIPS*, volume 25, pages 64–72, 2012.