A Hierarchical Bayesian Factorization Model for Implicit and Explicit Feedback Data

ThaiBinh Nguyen[1] and Atsuhiro Takasu[1,2]

[1] Department of Informatics,
SOKENDAI (The Graduate University for Advanced Studies), Tokyo, Japan
[2] National Institute of Informatics, Tokyo, Japan
{binh,takasu}@nii.ac.jp

Abstract. Matrix factorization (MF) is one of the most efficient methods for performing collaborative filtering. An MF-based method represents users and items by latent feature vectors that are obtained by decomposing the rating matrix of users to items. However, MF-based methods suffer from the cold-start problem: if no rating data are available for an item, the model cannot find a latent feature vector for that item, and thus cannot make a recommendation for it. In this paper, we present a hierarchical Bayesian model that can infer the latent feature vectors of items directly from the *implicit feedback* (e.g., clicks, views, purchases) when they cannot be obtained from the rating data. We infer the full posterior distributions of these parameters using a Gibbs sampling method. We show that the proposed method is strong with overfitting even if the model is very complex or the data are very sparse. Our experiments on real-world datasets demonstrate that our proposed method significantly outperforms competing methods on rating prediction tasks, especially for very sparse datasets.

Keywords: Collaborative filtering, item embedding, implicit feedback, explicit feedback, matrix factorization

1 Introduction

With the emergence of big data, recommender systems have become a core part of online services. The goal of a recommender system is to model user preferences by analyzing their history data and providing them with personalized recommendations. Collaborative filtering (CF) is an efficient approach for recommender systems that aims at predicting the rating of a user for an item given the past rating history of the users. Among various CF-based methods, matrix factorization (MF) is one of the most powerful approaches [11, 2].

MF-based algorithms represent user preferences and item attributes by latent feature vectors in a shared latent space. Typically, an MF-based algorithm finds the latent feature vectors of users and items by fitting the model with the observed ratings given in the user–item rating matrix [11, 12, 2, 15]. However, because each user can only rate a limited number of items, the rating matrix

is usually extremely sparse. Therefore, the performance of rating prediction declines if an item has too few ratings; or in an extreme case, if an item has no prior ratings, the system cannot learn its latent feature vector, and thus cannot recommend it to any user. This problem is referred to as the *cold-start* problem.

To address this problem, a common approach is to exploit other information about users and items, known as *side information* or *auxiliary information*. There are various types of side information, depending on the item being recommended (e.g., the genres of movies or text content of books). Such side information is successfully combined with traditional CF-based algorithms to alleviate the cold-start problem. For instance, in [14, 1, 16], text content was exploited for article recommendations; music content for song recommendations [9], or text content and category information for movie recommendations [10]. One limitation of these models is that such side information is not always available, or, in many cases, that information is available, but less informative for describing items (e.g., some items are described by a few keywords or very short texts).

This work focuses on exploiting another kind of feedback known as *implicit feedback* (e.g., clicks, views, purchases) as the auxiliary data. The advantage of using implicit feedback is that it is abundant and easily collected during the interactions of users with the system without requiring users to provide further interactions.

Related work. In [2], the authors proposed SVD++ which exploits implicit feedback to boost the performance of the original probabilistic matrix factorization (PMF) model [11]. However, in SVD++, implicit feedback is a binary matrix that indicates "who rates what", obtained by binarizing the rating data, an item has implicit feedback if and only if it has rating data; therefore, this model cannot model an item if it has no ratings.

Co-rating [5] combines explicit and implicit feedback in a unified framework. In the model, the explicit feedback are normalized into the range $[0, 1]$ and added to the implicit feedback matrix to form a unique matrix. This matrix is then factorized to obtain the latent feature vectors of users and items. In this way, the feature vector of an item can be inferred even if it does not have any rating data. The limitation of this method is that, after forming the final matrix, implicit and explicit feedback cannot be distinguished; therefore, this model cannot take into account the uncertainty of the implicit feedback.

In [13], the authors proposed a method for combining implicit and explicit feedback using expectation maximization (EM). To predict ratings for an item for which rating data are not available, the rating is inferred from ratings of its neighbors in terms of click data. However, the algorithm is based on an iterative EM-based algorithm in which the E-phase is a matrix factorization model. In other words, matrix factorization is performed multiple times and is therefore computationally expensive.

In [8], the authors proposed a probabilistic model for combining explicit and implicit feedback for making recommendations. In this model, the latent feature vector of an item for which rating data are not available can be learned

directly from the implicit feedback data. This is a combination of PMF [11] and an item embedding model; the model learns latent feature vectors for an item from the implicit feedback. In detail, the model consists of simultaneously factorizing the rating matrix and the positive point-wise mutual information (PPMI) matrix that is constructed from the click data. The item vectors are obtained by the factorization of the PPMI matrix and are then adjusted by the rating matrix. Although this model successfully combines implicit feedback data in learning latent feature vectors of items, it is prone to overfitting if the hyperparameters (i.e., the regularization parameters) are not tuned carefully. Usually, the hyperparameter tuning is very costly, especially when there are many hyperparameters, or when the data are large.

Present work. In this paper, we propose a fully hierarchical Bayesian treatment for the model proposed by Nguyen et al. [8]. In this model, instead of finding a point estimate for the model parameters, our method infers the full posterior distribution of these model parameters to capture their uncertainty. The missing ratings are predicted by integrating out the latent feature vectors of users and items. To this end, we place the Gaussian inverse Wishart priors on the mean vectors and covariance matrices of the latent feature vectors for the rating matrix and PPMI matrix [8]. We develop a Markov chain Monte Carlo (MCMC)-based method for inferring the full posterior distribution.

2 Preliminary

Suppose we have N users and M items. For each user–item pair (u, i), there can be two types of feedback: *explicit feedback* (also known as *rating data*) and *implicit feedback* (also known as *click data*). The rating data are represented by matrix $\mathbf{R} \in \mathbb{R}^{N \times M}$, in which element r_{ui} is the rating of user u for item i. r_{ui} can be a real number or a binary value (e.g., like/dislike). The click data are represented by a binary matrix $\mathbf{P} \in \{0, 1\}^{N \times M}$, where $p_{ui} = 1$ indicates that user u has clicked i at least once, and $p_{ui} = 0$ otherwise.

Generally, the rating matrix $\mathbf{R}$ is extremely sparse with many missing values (i.e., r_{ui} is not observed). We are interested in predicting these missing ratings.

2.1 Item embedding model according to implicit feedback data

Word embedding techniques have shown their success in many natural language processing tasks [6, 3]. By viewing each item in a recommender system as a word, the same assumptions that underlie word embedding models can be applied to modeling items. In [8], the authors proposed a method for an item embedding model based on implicit feedback data, i.e., a model that captures the relationship between items that are clicked by the same users.

In this item embedding model, item i is represented by two vectors: an *item vector* $\mathbf{w}_i$ and a *context vector* $\mathbf{z}_i$. The vectors have different roles: the item

vector describes the distribution of the item, and the context vector describes the distribution of the co-occurrence of an item with other items in its context.

In [8], the authors proposed an item-embedding scheme in which the *item vector* and the *context vector* are obtained by factorizing the PPMI matrix corresponding to the click data [8]:

$$PPMI(i,j) = \mathbf{w}_i^\top \mathbf{z}_i \tag{1}$$

The PPMI matrix is obtained by replacing the negative values by zeros in the point-wise mutual information (PMI) matrix. The elements of the PMI matrix are correlation measures for the co-occurrence of two items. Empirically, the PMI of items i and j can be approximated using the observed data:

$$PMI(i,j) = \log \frac{\#(i,j)|\mathcal{D}|}{\#(i)\#(j)}, \tag{2}$$

where $\#(i)$ and $\#(j)$ are the numbers of times items i and j are clicked, respectively. $\mathcal{D}$ is the set of item pairs that appear in the combined click history of all users. $\#(i,j)$ is the number of users who clicked both i and j.

2.2 Probabilistic model for implicit and explicit feedback data

After producing the item embedding model according to implicit feedback, Nguyen et al. [8] proposed a model that combines implicit and explicit feedback in a unified framework (PIE). PIE is a combination of the item-embedding model (described in Section 2.1) and the matrix factorization for rating data. In PIE, the latent feature vector $\mathbf{y}_i$ of item i is obtained by adding a small deviation $\mathbf{t}_i$ to the item vector $\mathbf{z}_i$. The graphical model of PIE is shown in Fig. 1a.

The main drawback of this model is that the parameter learning is a point estimation (MAP estimate), which is prone to overfitting when applying the trained model to unseen data. To avoid overfitting, we must tune the hyperparameters carefully. One approach is grid-search: we form a set of appropriate configurations of hyperparameters and train the model with these configurations. The configuration that produces the best performance on the validation set will be selected. However, in general, grid search is very costly, especially for large-scale data, or when the number of hyperparameters is large.

A straightforward way to avoid hyperparameter tuning is to introduce priors to the hyperparameters and optimize the log-posterior over both model parameters and hyperparameters. In this way, the hyperparameters will be learned from the data instead of tuned manually. However, this solution does not significantly improve the generalization of the model because it is still a point estimation and cannot capture the uncertainty of the model parameters.

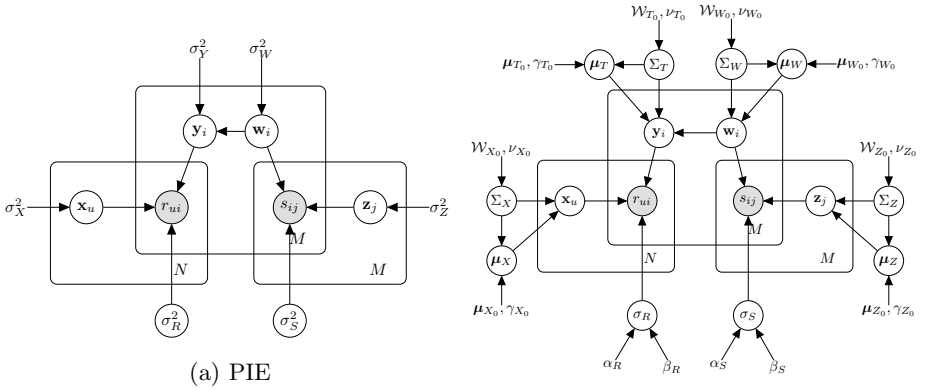
3 Proposed method

To address drawbacks described above, we propose a hierarchical, fully Bayesian model (HBFM) that can capture the uncertainty of the model parameters. In-

stead of approximating the posterior by its mode (the MAP estimate), we approximate the full posterior distributions of model parameters.

3.1 The model

We place the Gaussian inverse Wishart priors on the mean vectors and covariance matrices of the latent feature vectors. The graphical model is shown in Fig. 1b.



(a) PIE

(b) HPMF

Fig. 1: Graphical models of PIE [8] and HPMF (this paper). Because of space limitations, we omit the bias terms.

We assume that r_{ui} and s_{ij} are Gaussian distributions as follows:

$$p(r_{ui}|\mathbf{x}_u, \mathbf{y}_i, \theta_u, \rho_i, \sigma_R^2) = \mathcal{N}(r_{ui}|\mathbf{x}_u^\top \mathbf{y}_i + \eta_{ui}, \sigma_R^2) \tag{3}$$

$$p(s_{ui}|\mathbf{w}_i, \mathbf{z}_j, \sigma_S^2) = \mathcal{N}(s_{ij}|\mathbf{w}_i^\top \mathbf{z}_j, \sigma_S^2), \tag{4}$$

where θ_u and ρ_i are the biases of user u and item i, respectively; $\mathbf{y}_i = \mathbf{t}_i + \mathbf{w}_i$; $\eta_{ui} = \mu + \theta_u + \rho_i$; and μ is the global mean of the ratings.

The prior distributions of the latent feature vectors are assumed to be multivariate Gaussian distributions:

$$\begin{aligned} p(\mathbf{X}|\boldsymbol{\Theta}_X) &= \prod_{u=1}^{N} \mathcal{N}(\mathbf{x}_u|\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_X), \quad p(\mathbf{T}|\boldsymbol{\Theta}_T) = \prod_{i=1}^{M} \mathcal{N}(\mathbf{t}_i|\boldsymbol{\mu}_T, \boldsymbol{\Sigma}_T) \\ p(\mathbf{W}|\boldsymbol{\Theta}_W) &= \prod_{i=1}^{M} \mathcal{N}(\mathbf{w}_i|\boldsymbol{\mu}_W, \boldsymbol{\Sigma}_W), \quad p(\mathbf{Z}|\boldsymbol{\Theta}_Z) = \prod_{j=1}^{M} \mathcal{N}(\mathbf{z}_u|\boldsymbol{\mu}_Z, \boldsymbol{\Sigma}_Z) \end{aligned} \tag{5}$$

where $\boldsymbol{\mu}_X$, $\boldsymbol{\mu}_T$, $\boldsymbol{\mu}_W$, and $\boldsymbol{\mu}_Z$ are the mean vectors and $\boldsymbol{\Sigma}_X$, $\boldsymbol{\Sigma}_Y$, $\boldsymbol{\Sigma}_W$, and $\boldsymbol{\Sigma}_Z$ are the covariance matrices of $\mathbf{x}_u$, $\mathbf{y}_i$, $\mathbf{w}_i$, and $\mathbf{z}_j$, respectively; $\boldsymbol{\Theta}_X = \{\boldsymbol{\mu}_X, \Sigma_X\}$, $\boldsymbol{\Theta}_T = \{\boldsymbol{\mu}_T, \Sigma_T\}$, $\boldsymbol{\Theta}_W = \{\boldsymbol{\mu}_W, \Sigma_W\}$, and $\boldsymbol{\Theta}_Z = \{\boldsymbol{\mu}_Z, \Sigma_Z\}$.

To model the uncertainty of the latent feature vectors, we do not treat them as distributions of fixed hyperparameters. Instead, we further place Gaussian-inverse Wishart priors on $\mathbf{\Theta}_X$, $\mathbf{\Theta}_T$, $\mathbf{\Theta}_W$, and $\mathbf{\Theta}_Z$:

$$p(\mathbf{\Theta}_X|\mathbf{\Phi}_{X_0}) = \mathcal{N}(\boldsymbol{\mu}_X|\boldsymbol{\mu}_{X_0}, \boldsymbol{\Sigma}_X/\gamma_{X_0})\mathcal{W}^{-1}(\boldsymbol{\Sigma}_X|\mathcal{W}_{X_0}, \nu_{X_0}) \tag{6}$$

$$p(\mathbf{\Theta}_T|\mathbf{\Phi}_{T_0}) = \mathcal{N}(\boldsymbol{\mu}_T|\boldsymbol{\mu}_{T_0}, \boldsymbol{\Sigma}_T/\gamma_{T_0})\mathcal{W}^{-1}(\boldsymbol{\Sigma}_T|\mathcal{W}_{T_0}, \nu_{T_0}) \tag{7}$$

$$p(\mathbf{\Theta}_W|\mathbf{\Phi}_{W_0}) = \mathcal{N}(\boldsymbol{\mu}_W|\boldsymbol{\mu}_{W_0}, \boldsymbol{\Sigma}_W/\gamma_{W_0})\mathcal{W}^{-1}(\boldsymbol{\Sigma}_W|\mathcal{W}_{W_0}, \nu_{W_0}) \tag{8}$$

$$p(\mathbf{\Theta}_Z|\mathbf{\Phi}_{Z_0}) = \mathcal{N}(\boldsymbol{\mu}_Z|\boldsymbol{\mu}_{Z_0}, \boldsymbol{\Sigma}_Z/\gamma_{Z_0})\mathcal{W}^{-1}(\boldsymbol{\Sigma}_Z|\mathcal{W}_{Z_0}, \nu_{Z_0}), \tag{9}$$

where: $\mathbf{\Phi}_{X_0} = \{\boldsymbol{\mu}_{X_0}, \gamma_{X_0}, \mathcal{W}_{X_0}, \nu_{X_0}\}$, $\mathbf{\Phi}_{T_0} = \{\boldsymbol{\mu}_{T_0}, \gamma_{T_0}, \mathcal{W}_{T_0}, \nu_{T_0}\}$, $\mathbf{\Phi}_{W_0} = \{\boldsymbol{\mu}_{W_0}, \gamma_{W_0}, \mathcal{W}_{W_0}, \nu_{W_0}\}$, and $\mathbf{\Phi}_{Z_0} = \{\boldsymbol{\mu}_{Z_0}, \gamma_{Z_0}, \mathcal{W}_{Z_0}, \nu_{Z_0}\}$.

Here, $\mathcal{W}^{-1}$ is the inverse Wishart distribution with ν_0 degrees of freedom and a $d \times d$ scaling matrix $\mathcal{W}_0$:

$$\mathcal{W}^{-1}(\boldsymbol{\Sigma}|\mathcal{W}_0, \nu_0) = \frac{1}{C}|\boldsymbol{\Sigma}|^{-(\nu_0-d-1)/2}\exp(-\frac{1}{2}Tr(\mathcal{W}_0\boldsymbol{\Sigma}^{-1})), \tag{10}$$

where C is a normalizing constant and $Tr(.)$ is the trace of a matrix.

The Gaussian inverse Wishart prior is adopted because it is the conjugate prior of the multivariate Gaussian distribution. This selection of the prior allows the conditional distributions derived from the posterior distributions to be sampled easily. Similarly, we place inverse Gamma priors [17] on the variance σ_R^2:

$$p(\sigma_R^2|\alpha_R, \beta_R) \quad = \quad IG(\sigma_R^2|\alpha_R, \beta_R), \tag{11}$$

where $IG(.)$ is the inverse Gamma distribution [17]:

$$IG(x|\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}x^{-\alpha-1}exp(-\frac{\beta}{x}) \tag{12}$$

Choosing the inverse Gamma distribution, which is the conjugate prior of the variance of a Gaussian distribution, makes it easy to sample from the posterior distribution. Indeed, this distribution has also been proven to model the unknown variance of a Gaussian distribution effectively [17].

We place Gaussian priors over the bias terms as follows.

$$p(\theta_u|\sigma_\theta^2) = \mathcal{N}(\theta_u|0, \sigma_\theta^2), \quad p(\rho_i|\sigma_\rho^2) = \mathcal{N}(\rho_i|0, \sigma_\rho^2), \tag{13}$$

where, σ_θ^2 and σ_ρ^2 are inverse Gamma distributions [17]:

$$p(\sigma_\theta^2|\alpha_\theta, \beta_\theta) = IG(\sigma_\theta^2|\alpha_\theta, \beta_\theta), \quad p(\sigma_\rho^2|\alpha_\rho, \beta_\rho) = IG(\sigma_\rho^2|\alpha_\rho, \beta_\rho) \tag{14}$$

We place an inverse Gamma [17] prior on the variance of σ_S^2 of r_{ij}:

$$p(\sigma_S^2|\alpha_S, \beta_S) \quad = \quad IG(\sigma_S^2|\alpha_S, \beta_S) \tag{15}$$

3.2 Posterior inference

Our goal is to find the posterior distribution of the model parameters. The posterior distribution is analytically intractable, so we employ MCMC-based methods, which are widely used for approximating distributions [7]. The key idea of these methods is to construct a Markov chain that converges to the posterior distribution of the model. Each state of the Markov chain is a set of model parameters. The posterior distribution is characterized by the samples from that Markov chain. In this paper, we use Gibbs sampling [7], a kind of MCMC that alternatively samples each variable conditioned on the remaining variables.

Sampling $\mathbf{x}_u$, $\mathbf{t}_i$, $\mathbf{w}_i$, and $\mathbf{z}_j$. The conditional distribution over the user latent feature vector $\mathbf{x}_u$, conditioned on the observed ratings, the latent feature vectors of items, and the hyperparameters, is Gaussian:

$$\begin{aligned} p(\mathbf{x}_u|\mathbf{R},\mathbf{Y},\boldsymbol{\mu}_X,\boldsymbol{\theta},\boldsymbol{\rho},\boldsymbol{\Sigma}_X) &= \mathcal{N}(\mathbf{x}_u|\boldsymbol{\mu}^*_{X_u},\boldsymbol{\Sigma}^*_{X_u}) \\ &\propto p(\mathbf{x}_u|\boldsymbol{\mu}_X,\boldsymbol{\Sigma}_X)\prod_{i\in\mathcal{R}_u}\mathcal{N}(r_{ui}|\mathbf{x}_u^\top\mathbf{y}_i+\eta_{ui},\sigma_R^2), \end{aligned} \tag{16}$$

where $\eta_{ui}=\theta_u+\rho_i+\mu$, $\boldsymbol{\theta}=\{\theta_u\}_{u=1}^N$, $\boldsymbol{\rho}=\{\rho_i\}_{i=1}^M$, and

$$\boldsymbol{\Sigma}^*_{X_u} = \Big(\boldsymbol{\Sigma}_X^{-1}+\frac{1}{\sigma_R^2}\sum_{i\in\mathcal{R}_u}\mathbf{y}_i\mathbf{y}_i^\top\Big)^{-1} \tag{17}$$

$$\boldsymbol{\mu}^*_{X_u} = \boldsymbol{\Sigma}^*_{X_u}\Big[\boldsymbol{\Sigma}_X^{-1}\boldsymbol{\mu}_X+\frac{1}{\sigma_R^2}\sum_{i\in\mathcal{R}_u}(r_{ui}-\eta_{ui})\mathbf{y}_i\Big]. \tag{18}$$

Similarly, we can obtain the posterior distribution of $\mathbf{t}_i$, $\mathbf{w}_i$ and $\mathbf{z}_j$.

Sampling $\boldsymbol{\Theta_X}=\{\boldsymbol{\mu_X},\boldsymbol{\Sigma_X}\}$, $\boldsymbol{\Theta_T}=\{\boldsymbol{\mu_T},\boldsymbol{\Sigma_T}\}$, $\boldsymbol{\Theta_W}=\{\boldsymbol{\mu_W},\boldsymbol{\Sigma_W}\}$, and $\boldsymbol{\Theta_Z}=\{\boldsymbol{\mu_Z},\boldsymbol{\Sigma_Z}\}$. The posterior distribution over $\boldsymbol{\Theta}_X=\{\boldsymbol{\mu}_X,\boldsymbol{\Sigma}_X\}$ conditioned on user latent feature vectors and $\boldsymbol{\Phi}_{X_0}=\{\boldsymbol{\mu}_{X_0},\gamma_{X_0},\mathcal{W}_{X_0},\nu_{X_0}\}$ is a Gaussian inverse Wishart distribution:

$$p(\boldsymbol{\mu}_X,\boldsymbol{\Sigma}_X|\mathbf{X},\boldsymbol{\Phi}_{X_0}) = \mathcal{N}(\boldsymbol{\mu}_X|\boldsymbol{\mu}^*_{X_0},\boldsymbol{\Sigma}_X/\gamma^*_{X_0})\mathcal{W}^{-1}(\boldsymbol{\Sigma}_X|\mathcal{W}^*_{X_0},\nu^*_{X_0}) \tag{19}$$

$$\propto p(\mathbf{X}|\boldsymbol{\mu}_X,\boldsymbol{\Sigma}_X)p(\boldsymbol{\mu}_X,\boldsymbol{\Sigma}_X|\boldsymbol{\Phi}_{X_0}), \tag{20}$$

where:

$$\boldsymbol{\mu}^*_{X_0} = \frac{\gamma_{X_0}\boldsymbol{\mu}_{X_0}+N\bar{\mathbf{x}}}{\gamma_{X_0}+N}, \quad \gamma^*_{X_0}=\gamma_{X_0}+N, \nu^*_{X_0}=\nu_{X_0}+N \tag{21}$$

$$\mathcal{W}^*_{X_0} = \mathcal{W}_{X_0}+N\bar{\mathbf{S}}+\frac{\gamma_{X_0}N}{\gamma_{X_0}+N}(\boldsymbol{\mu}_{X_0}-\bar{\mathbf{x}})(\boldsymbol{\mu}_{X_0}-\bar{\mathbf{x}})^\top \tag{22}$$

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{u=1}^N\mathbf{x}_u, \quad \bar{\mathbf{S}} = \frac{1}{N}\sum_{u=1}^N\mathbf{x}_u\mathbf{x}_u^\top \tag{23}$$

Similarly, we can obtain the posterior distributions over $\boldsymbol{\Theta}_T$, $\boldsymbol{\Theta}_W$, and $\boldsymbol{\Theta}_Z$ using exactly the same form.

Sampling bias terms $\boldsymbol{\theta_u}$ and $\boldsymbol{\rho_i}$. The posterior distribution over the user bias term θ_u is Gaussian:

$$\begin{aligned} p(\theta_u|\mathbf{R}, \mathbf{X}, \mathbf{Y}, \boldsymbol{\rho}, \sigma_R^2) &= \mathcal{N}(\theta_u|\xi_u^*, (\sigma_{\theta_u}^*)^2 \\ &\propto p(\mathbf{R}|\mathbf{X}, \mathbf{Y}, \boldsymbol{\rho}, \sigma_R^2) p(\theta_u|\sigma_\theta^2), \end{aligned} \tag{24}$$

where:

$$(\sigma_{\theta_u}^*)^2 = \Big(\frac{1}{\sigma_\theta^2} + \frac{|\mathcal{R}_u|}{\sigma_R^2}\Big)^{-1}, \quad \xi_u^* = \Big(\frac{\sigma_{\theta_u}^*}{\sigma_R}\Big)^2 \sum_{i \in \mathcal{R}_u} \Big[r_{ui} - (\mu + \rho_i + \mathbf{x}_u^\top \mathbf{y}_i)\Big] \tag{25}$$

The posterior distribution over the ρ_i can be obtained using the same form.

Sampling $\boldsymbol{\sigma_R^2}$ and $\boldsymbol{\sigma_S^2}$. The posterior distribution over σ_R^2, conditioned on the rating data, user latent factor matrix $\mathbf{X}$, item latent factor matrix $\mathbf{Y}$, and bias matrices $\boldsymbol{\theta}$, $\boldsymbol{\rho}$, is given as:

$$\begin{aligned} p(\sigma_R^2|\mathbf{R}, \mathbf{X}, \mathbf{Y}, \alpha_R, \beta_R) &= IG(\sigma_R^2|\alpha_R^*, \beta_R^*) \\ &\propto p(\mathbf{R}|\mathbf{X}, \mathbf{Y}, \boldsymbol{\theta}, \boldsymbol{\rho}, \sigma_R^2) p(\sigma_R^2|\alpha_R, \beta_R) \end{aligned} \tag{26}$$

where:

$$\alpha_R^* = \alpha_R + \frac{|\mathcal{R}|}{2}, \quad \beta_R^* = \beta_R + \frac{1}{2} \sum_{(i,j) \in \mathcal{R}} \Big[r_{ui} - (\mathbf{x}_u^\top \mathbf{y}_i + \eta_{ui})\Big]^2 \tag{27}$$

The conditional distribution over σ_S^2 can be obtained using the same form.

Sampling $\boldsymbol{\sigma_\theta^2}$ and $\boldsymbol{\sigma_\rho^2}$. The conditional distribution over σ_θ^2 conditioned on the bias terms of users is an inverse Gamma distribution:

$$p(\sigma_\theta^2|\boldsymbol{\theta}, \alpha_\theta, \beta_\theta) = IG(\sigma_\theta^2|\alpha_\theta^*, \beta_\theta^*) \quad \propto \quad p(\boldsymbol{\theta}|\sigma_\theta^2) p(\sigma_\theta^2|\alpha_\theta, \beta_\theta), \tag{28}$$

where:

$$\alpha_\theta^* = \alpha_\theta + \frac{N}{2}, \quad \beta_\theta^* = \beta_\theta + \frac{1}{2} \sum_{u=1}^{N} \theta_u^2 \tag{29}$$

The conditional distribution over σ_ρ^2 conditioned on the bias terms of items can be obtained using the same form.

Computational complexity. From the formulas for posterior distribution sampling, we can observe that the most expensive computations lie in the sampling of the latent feature vectors ($\mathbf{x}_u$, $\mathbf{t}_i$, $\mathbf{w}_i$ and $\mathbf{z}_j$), which require computing the inverses of matrices. It is easy to show that in each iteration, the complexity for sampling the latent feature vectors of N users (matrix $\mathbf{X}$) is $\mathcal{O}(d^2|\mathcal{R}| + d^3 N)$. Similarly, the complexities for sampling matrix $\mathbf{T}$, $\mathbf{W}$, and $\mathbf{Z}$ are $\mathcal{O}(d^2|\mathcal{R}| + d^3 N)$, $\mathcal{O}(d^2|\mathcal{S}| + d^3 M)$, and $\mathcal{O}(d^2|\mathcal{S}| + d^3 M)$, respectively, where $|\mathcal{R}|$ and $|\mathcal{S}|$ are the numbers of observed ratings and observed clicks, respectively. However, note that the posterior distribution of $\mathbf{x}_u$ does not depend on other users; therefore, the sampling of matrix $\mathbf{X}$ can be performed efficiently in parallel. Similarly, sampling $\mathbf{T}$, $\mathbf{W}$, and $\mathbf{Z}$ can also be sped up by performing them in parallel.

3.3 Rating prediction

The posterior predictive distribution of the unseen rating value $\hat{r}_{ui}$ of item i by user u is obtained by integrating out the model parameters and hyperparameters:

$$p(\hat{r}_{ui}|\mathcal{O}) = \int \cdots \int p(\hat{r}_{ui}|\boldsymbol{\Omega})p(\boldsymbol{\Omega})d\{\boldsymbol{\Omega}\}, \tag{30}$$

where $\mathcal{O}$ is the observed data and $\boldsymbol{\Omega}$ is the set of all parameters.

The above posterior predictive distribution is analytically intractable, so we approximate it by sampling the parameters using the Gibbs sampling described in Section 3.2. The predicted rating value can be approximated as follows:

$$\begin{aligned} p(\hat{r}_{ui}|\mathcal{O}) &\approx \frac{1}{K}\sum_{k=1}^{K} p(\hat{r}_{ui}|\mathbf{x}_u^{(k)}, \mathbf{y}_i^{(k)}, \theta_u^{(k)}, \rho_i^{(k)}, (\sigma_R^2)^{(k)}) \\ &= \frac{1}{K}\sum_{k=1}^{K} \mathcal{N}\Big(\hat{r}_{ui}|\eta_{ui}^{(k)} + {\mathbf{x}_u^{(k)}}^{\top}\mathbf{y}_i^{(k)}, (\sigma_R^2)^{(k)}\Big), \end{aligned} \tag{31}$$

where K is the number of samples taken from the posterior distribution, $(.)^{(k)}$ is the kth sample, and $\eta_{ui}^{(k)} = \mu + \theta_u^{(k)} + \rho_i^{(k)}$.

We consider two rating prediction tasks: (i) **in-matrix** prediction: predict the rating by user u of item i, where i has not been rated by u but has been rated by at least one other user (i.e., i appears at least once in the training set of the rating data); and (ii) **out-matrix** prediction: predict the rating by user u of item i, where i has not been rated by any user (i.e., i does not appear in the training set of the rating data).

In Eq. 31, $\mathbf{y}_i^{(k)} = \mathbf{w}_i^{(k)} + \mathbf{t}_i^{(k)}$ for the *in-matrix* prediction task; $\mathbf{y}_i^{(k)} = \mathbf{w}_i^{(k)}$ and $\eta_{ui}^{(k)} = \mu + \theta_u^{(k)}$ for the *out-matrix* prediction task.

4 Empirical study

4.1 Datasets

Data description. We used three public datasets of different domains with varying sizes. (1) **MovieLens 1M** (ML-1m): a dataset of user-movie ratings collected from MovieLens, an online film service. It contains 1 million ratings in the range $1-5$ of 4000 movies by 6000 users. This dataset is available at GroupLens[3]. (2) **MovieLens 20M** (ML-20m): another dataset of user-movie ratings collected from MovieLens. It contains 20 million ratings in the range $1-5$ of 27,000 movies by 138,000 users. This dataset is available at GroupLens[4]. (3) **Bookcrossing**: A dataset collected in August and September 2004 from the Book-Crossing website[5]. This dataset contains 278,858 users (anonymized but

[3] https://grouplens.org/datasets/movielens/1m/
[4] https://grouplens.org/datasets/movielens/20m/
[5] http://www.bookcrossing.com/

with demographic information) providing 1,149,780 ratings (explicit/implicit) of 271,379 books. We removed users and items that had no explicit feedback.

The MovieLens datasets contain only rating data, so we employed a preprocess phase to obtain the click data. We binarized the original rating data and interpreted it as click data. Furthermore, because rating data are only a small part of the click data, we randomly selected from original ratings with different percentages, assuming that only these amounts of ratings were available. Details of these datasets after preprocessing are shown in Table 1.

Table 1: Datasets obtained by selecting ratings from the original ratings of MovieLens datasets with different percentages

Picked from ML1-20			Picked from ML20-20		
Dataset	% rating picked	Density of rating matrix (%)	Dataset	% rating picked	Density of rating matrix (%)
ML1-10	10	0.3561	ML20-10	10	0.0836
ML1-20	20	0.6675	ML20-20	20	0.1001
ML1-50	50	1.6022	ML20-50	50	0.2108

4.2 Experimental protocol

We used the click data and 80% of the rating data to train the model; the remaining 20% of the rating data was used as the test data to evaluate the model. In evaluating the *in-matrix* prediction task, when splitting data, we made sure that every item in the test set appeared at least once in the training set. In evaluating the *out-matrix* prediction task, we made sure that none of the items in the test set appeared in the training set (to ensure that none of the items in the test set had any previous ratings).

Evaluation metric. We used Root Mean Square Error (RMSE) as the metric to measure the performance of the models. RMSE measures the deviation between the rating predicted by the model and the true rating (given by the test set), and is defined as follows.

$$RMSE = \sqrt{\frac{1}{|Test|} \sum_{(u,i) \in Test} (r_{ui} - \hat{r}_{ui})^2}, \tag{32}$$

where $|Test|$ is the size of the test set.

Competing methods. For the **in-matrix** prediction task, we compared our method with the following baseline methods:

1. *PMF* [11]: a state-of-the-art method for rating predictions
2. *BPMF* [12]: the Bayesian treatment of PMF [11]
3. *NMF* (non-negative matrix factorization) [4]: a matrix factorization method which requires the components of user and item factors to be non-negative
4. *PIE* [8]: the model described in Section 2.2
5. *SVD++* [2]: a factor model that exploits both explicit and implicit feedback in rating predictions

For the **out-matrix** prediction task, we compared our proposed method with PIE [8], which is described in Section 2.2.

Parameter settings. We varied the dimension of the latent space ($d = 20, 30, 50,$ 100) to study the performance of the models with respect to the dimensionality of the latent feature vectors.

For PMF, NMF and SVD++, we used grid search to find the optimal values of the hyperparameters that produced the best performance on a validation set. For the PIE model [8], we fixed $\lambda = 1$ and used grid search to find the optimal values of the remaining parameters that gave good performance on the validation set. For BPMF [12], hyperparameters were set following the original paper.

Regarding our proposed method, HBFM, for simplicity, we set the parameters as follows: $\mathcal{W}_{\mathcal{F}} = \mathbf{I}_d$, $\nu_{\mathcal{F}_0} = d$, $\gamma_{\mathcal{F}_0} = 1$, and $\mu_{\mathcal{F}_\mathbf{0}} = \mathbf{0}$ ($\mathcal{F} = \{X, T, W, Z\}$). We adopted uninformative priors for the noise variances; therefore, we set the hyperparameters for the inverse Gamma distributions as follows: $\alpha_R = \alpha_S = \alpha_\theta = \alpha_\rho = 0$ and $\beta_R = \beta_S = \beta_\theta = \beta_\rho = 0$. For the Gibbs sampling process, we ignored the first 1000 samples as "burn-in". The following 100 samples were selected to approximate the posterior distributions.

4.3 Results

We report the RMSEs on the test datasets for the *in-matrix* and *out-matrix* prediction tasks in Table 2 and Table 3, respectively. We can see that HBFM outperformed the competing methods for all values of d.

For small values of d (e.g., $d = 20, 50$), PIE and HP-PIE perform better than the other methods, indicating the effectiveness of exploiting click data in boosting the performance of rating predictions. When d exceeds 150, the test RMSEs for PMF, NMF, SVD++, and PIE tend to increase, whereas those for BPMF and HBFM continue to decrease. This is because when d increases, the number of parameters increases and the models become more complex. PMF, NMF, SVD++, and PIE do not handle the complexity of the model well; therefore, they tend to overfit. By contrast, BPMF and HBFM, which can manage the complexity of the models well, continue improving the test RMSEs. This shows that the full Bayesian model that can manage the uncertainty of the model parameters is an effective approach for avoiding overfitting.

Table 2: Test RMSEs for different numbers of latent features

(a) ML1-20 dataset

Methods	# of latent features d				
	20	50	100	150	200
PMF	1.0053	0.9941	0.9574	0.9628	0.9715
NMF	0.9971	0.9734	0.9571	0.9605	0.9711
SVD++	0.9464	0.9342	0.9023	0.9148	0.9235
BPMF	0.9339	0.9191	0.8971	0.8824	0.8731
PIE	0.9218	0.9021	0.8911	0.9013	0.9125
HBFM	**0.9012**	**0.8834**	**0.8617**	**0.8594**	**0.8512**

(b) ML20-20 dataset

Methods	# of latent features d				
	20	50	100	150	200
PMF	0.9627	0.9098	0.8832	0.8901	0.9015
NMF	0.8988	0.8927	0.8856	0.8942	0.9031
SVD++	0.8947	0.8655	0.8532	0.8598	0.8641
BPMF	0.8804	0.8576	0.8462	0.8397	0.8301
PIE	0.8788	0.8532	0.8474	0.8501	0.8602
HBFM	**0.8521**	**0.8401**	**0.8325**	**0.8245**	**0.8189**

(c) Bookcrossing dataset

Methods	# of latent features d				
	20	50	100	150	200
PMF	2.0231	2.0105	1.9834	1.9921	1.9989
NMF	1.9477	1.9132	1.9092	1.9132	1.9198
SVD++	1.8090	1.7968	1.7729	1.7823	1.7891
BPMF	1.7941	1.7873	1.7728	1.7693	1.7601
PIE	1.6704	1.6501	1.6341	1.6401	1.6487
HBFM	**1.6623**	**1.6431**	**1.6028**	**1.5931**	**1.5867**

Table 3: Test RMSEs for the out-matrix prediction task

# of	ML1-20		ML20-20		Bookcrossing	
features	PIE	HBFM	PIE	HBFM	PIE	HBFM
20	1.0066	**0.9902**	0.9686	**0.9523**	1.7257	**1.7028**
50	1.0062	**0.9811**	0.9436	**0.9357**	1.6484	**1.6245**
100	1.0044	**0.9801**	0.9374	**0.9211**	1.6398	**1.6201**
150	1.0089	**0.9758**	0.9403	**0.9188**	1.6405	**1.6178**
200	1.0132	**0.9695**	0.9489	**0.9101**	1.6497	**1.6102**

Impact of the sparsity of the dataset on the methods. We studied the effectiveness of the proposed method for datasets with different levels of sparsity by training models with the ML1-10, ML1-20, ML1-50, ML20-10, ML20-20 and ML20-50 datasets. The test RMSEs are shown in Table 4.

Table 4: Test RMSEs for datasets with different levels of sparsity. The dimensionality of feature vectors is fixed: $d = 20$

(a) In-matrix prediction

Method	ML1m			ML20m		
	ML1-10	ML1-20	ML1-50	ML20-10	ML20-20	ML20-50
PMF	1.0471	0.9941	0.9574	0.9627	0.9098	0.8532
NMF	1.0179	0.9734	0.9571	0.8988	0.8927	0.8856
SVD++	0.9757	0.9342	0.9023	0.8947	0.8655	0.8489
BPMF	0.9364	0.9191	0.8971	0.8804	0.8576	0.8362
PIE	0.9318	0.9021	0.8801	0.8788	0.8532	0.8474
HBFM	**0.9012**	**0.8834**	**0.8617**	**0.8521**	**0.8401**	**0.8325**

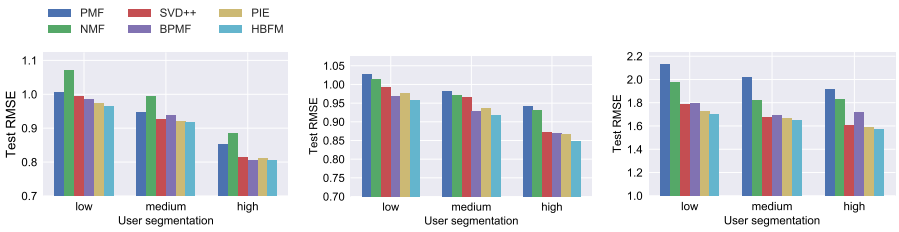
(b) Out-matrix prediction

Method	ML1m			ML20m		
	ML1-10	ML1-20	ML1-50	ML20-10	ML20-20	ML20-50
PIE	1.0376	1.0066	0.9961	0.9762	0.9686	0.9601
HBFM	**1.0231**	**0.9913**	**0.9728**	**0.9634**	**0.9521**	**0.9489**

We can observe that denser rating data improved test RMSE values for all methods. This is reasonable because when more rating data are available for training, the prediction is more accurate. When the data are extremely sparse (e.g., ML1-10 or ML20-10), although managing the complexity of the model for sparse data is a challenging task, PIE and HBFM perform better than the other methods because they leverage the sparsity of rating data by the click data. For all settings, HBFM outperforms the competing methods. These results clearly show the effectiveness of exploiting click data and managing the complexity of sparse datasets.

Performance for different segmentations of users. We further test the effectiveness of our method with different segments of users. We divided users into three segments based on the number of items for which they had provided ratings, and compared the performances of the methods for each group. These segments are: (i) *low*: users who provide fewer than 20 ratings; (ii) *medium*: users who provide fewer than 50 and more than 20 ratings; and (iii) *high*: users who provide 50 or more ratings.

The test RMSEs in Fig. 2 show that our method (HBFM) outperforms all competing methods for all user segments for the three datasets. From the results,



(a) ML1-20 dataset (b) ML20-20 dataset (c) Bookcrossing dataset

Fig. 2: Test RMSEs for different segmentations of users

we can also see that all the methods perform better when more explicit feedback is provided. This is reasonable because explicit feedback is much more reliable than implicit feedback for inferring users' preferences.

5 Discussion and future work

In this paper, we have proposed HBFM, a fully Bayesian model that combines explicit and implicit feedback to address the cold-start problem in collaborative filtering. This is a Bayesian treatment of the PIE model [8], in which priors are placed on the hyperparameters such as the covariance matrix of latent feature vectors or the variance of rating data. We developed a Gibbs sampling-based method to approximate the posterior distributions over latent feature vectors of users and items. The experiments show that HBFM provides good control over the capacity, and can be applied to models with large numbers of parameters and very sparse data.

Several future directions are possible. One is to make the model more flexible by developing a nonparametric algorithm that can efficiently find the appropriate dimensionality of latent feature vectors instead of empirically tuning the method. Another direction is to generalize the model to adopt different types of explicit feedback. In the present model, we assumed that the rating data were random variables with Gaussian distributions. This model may not work well when the data are binary feedback (e.g., like/dislike, purchase/not purchase); in that case, a Bernoulli distribution model may be more suitable.

Acknowledgments. This work was supported by a JSPS Grant-in-Aid for Scientific Research (B) (15H02789, 15H02703).

References

1. Gopalan, P.K., Charlin, L., Blei, D.: Content-based recommendations with poisson factorization. In: Advances in Neural Information Processing Systems 27. pp. 3176–3184 (2014)

2. Koren, Y.: Factorization meets the neighborhood: A multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 426–434 (2008)
3. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: Proceedings of the 31st International Conference on Machine Learning. pp. 1188–1196 (2014)
4. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. In: Advances in Neural Information Processing Systems 13. pp. 556–562 (2001)
5. Liu, N.N., Xiang, E.W., Zhao, M., Yang, Q.: Unifying explicit and implicit feedback for collaborative filtering. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management. pp. 1445–1448 (2010)
6. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th International Conference on Neural Information Processing Systems. pp. 3111–3119 (2013)
7. Neal, R.M.: Probabilistic inference using Markov chain Monte Carlo methods. Tech. Rep. CRG-TR-93-1, Dept. of Computer Science, University of Toronto (1993)
8. Nguyen, T., Aihara, K., Takasu, A.: A probabilistic model for collaborative filtering with implicit and explicit feedback data. CoRR abs/1705.02085 (2017), `http://arxiv.org/abs/1705.02085`
9. Oord, A.v.d., Dieleman, S., Schrauwen, B.: Deep content-based music recommendation. In: Proceedings of the 26th International Conference on Neural Information Processing Systems. pp. 2643–2651 (2013)
10. Park, S., Kim, Y.D., Choi, S.: Hierarchical bayesian matrix factorization with side information. In: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence. pp. 1593–1599 (2013)
11. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: Proceedings of the 20th International Conference on Neural Information Processing Systems. pp. 1257–1264 (2007)
12. Salakhutdinov, R., Mnih, A.: Bayesian probabilistic matrix factorization using markov chain monte carlo. In: Proceedings of the 25th International Conference on Machine Learning. pp. 880–887 (2008)
13. Wang, B., Rahimi, M., Zhou, D., Wang, X.: Expectation-maximization collaborative filtering with explicit and implicit feedback. In: Advances in Knowledge Discovery and Data Mining: 16th Pacific-Asia Conference, PAKDD 2012, Proceedings, Part I. pp. 604–616 (2012)
14. Wang, C., Blei, D.M.: Collaborative topic modeling for recommending scientific articles. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 448–456 (2011)
15. Wang, H., SHI, X., Yeung, D.Y.: Collaborative recurrent autoencoder: Recommend while learning to fill in the blanks. In: Advances in Neural Information Processing Systems 29. pp. 415–423 (2016)
16. Wang, H., Wang, N., Yeung, D.Y.: Collaborative deep learning for recommender systems. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1235–1244 (2015)
17. Witkovsky, V.: Computing the distribution of a linear combination of inverted gamma variables. Kybernetika 37(1), 79–90 (2001)