Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Sparse Probabilistic Matrix Factorization by Laplace Distribution for Collaborative Filtering

Liping Jing, Peng Wang, Liu Yang
Beijing Key Lab of Traffic Data Analysis and Mining, Beijing Jiaotong University
Beijing, China
{lpjing,wangpeng,11112091}@bjtu.edu.cn

Abstract

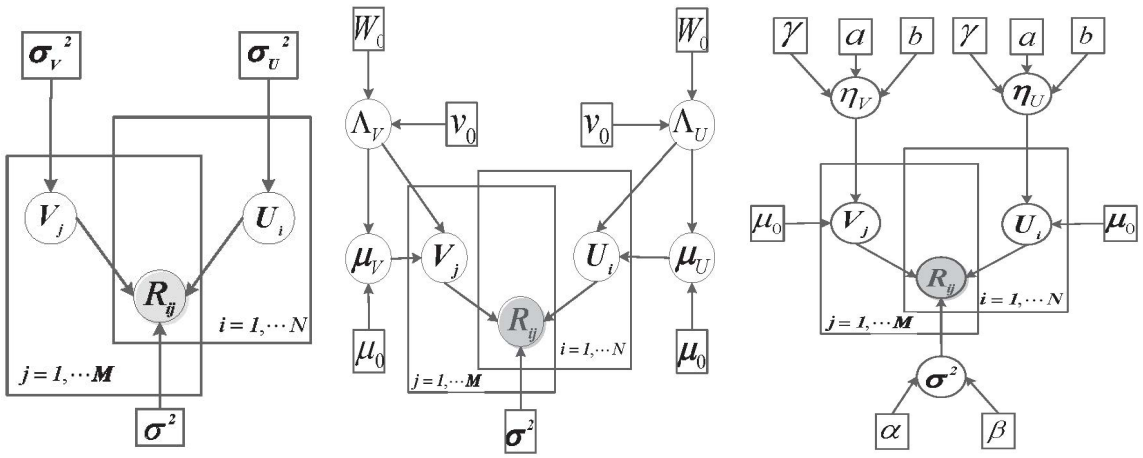
In recommendation systems, probabilistic matrix factorization (PMF) is a state-of-the-art collaborative filtering method by determining the latent features to represent users and items. However, two major issues limiting the usefulness of PMF are the sparsity problem and long-tail distribution. Sparsity refers to the situation that the observed rating data are sparse, which results in that only part of latent features are informative for describing each item/user. Long tail distribution implies that a large fraction of items have few ratings. In this work, we propose a sparse probabilistic matrix factorization method (SPMF) by utilizing a Laplacian distribution to model the item/user factor vector. Laplacian distribution has ability to generate sparse coding, which is beneficial for SPMF to distinguish the relevant and irrelevant latent features with respect to each item/user. Meanwhile, the tails in Laplacian distribution are comparatively heavy, which is rewarding for SPMF to recommend the tail items. Furthermore, a distributed Gibbs sampling algorithm is developed to efficiently train the proposed sparse probabilistic model. A series of experiments on Netflix and Movielens datasets have been conducted to demonstrate that SPMF outperforms the existing PMF and its extended version Bayesian PMF (BPMF), especially for the recommendation of tail items.

1 Introduction

With the emerging of big data, recommender systems play a more and more important role to provide personalized recommendation and improve the user experience [Adomavicius and Tuzhilin, 2005]. In the last decade, the collaborative filtering (CF) methods based on matrix factorization (MF) [Koren *et al.*, 2009] have shown their ability to build accurate prediction models and are widely adopted in commercial world such as Amazon, Google and Netflix [Dror *et al.*, 2012]. MF methods predict the user preference by determining the latent features of users and items. More precisely, MF method factorizes the rating matrix into two low-rank matrices, one for latent user factor and the other for latent item factor. By multiplying these two latent factors, we can complete the original incomplete rating matrix and do recommendation.

Even though a variety of MF methods [Berry *et al.*, 1999; Salakhutdinov and Mnih, 2007; 2008; Chen *et al.*, 2011; Shi *et al.*, 2013; Bauer and Nanopoulos, 2014] have been proposed and lead to promising recommendation results, they are limited in modern recommendation systems due to the sparse and large-scale rating data. The sparsity of data refers to the lack of observed rating data that makes it difficult and unreliable to predict the user preference. One accompanying phenomenon is long-tail effect where a large fraction of items have few ratings [Anderson, 2006]. Although the amount of users relating to each individual tail item is small in absolute numbers, collectively they cover a substantial fraction of all users. Additionally, a user's rarer purchases in e-commerce are also more informative of their tastes than their purchases of popular items. Hence, making use of the tail items is important to predict the user preference in modern recommendation systems. In such sparse observed rating data, the number of items that each user are interested in is very small. Recall the principle of MF methods, each user factor measures how much the user likes items that score high on the corresponding item factor [Koren *et al.*, 2009]. Thus, each user can be characterized well by a few latent features, i.e., the user factor vector in latent feature space should be sparse. This is the main motivation of our work.

For the sake of dealing with sparse rating data, probabilistic matrix factorization (PMF) [Salakhutdinov and Mnih, 2007] was proposed based on the assumption that the latent factors follow Gaussian distribution. This assumption makes sense when the observations are continuous, however it is less justified when the data are on an ordinal scale [Lakshminarayananw *et al.*, 2011]. Later, Bayesian matrix factorization (BPMF) [Salakhutdinov and Mnih, 2008] is estimated by imposing Gaussian-Wishart priors over the factor vectors and tries to generate a non-Gaussian distribution via a product of two multivariate Gaussians. Recently, a sparse covariance prior is adopted in [Shi *et al.*, 2013] to enforce the user and item factors and make each latent feature reflect the semantics more properly. To date, PMF and its variants become the arguably representative CF methods, but they can not guarantee that each user is effectively represented by the most informative latent features. In real applications, obtaining proper



(a) PMF (b) BPMF (c) SPMF

Figure 1: Graphical models for PMF, BPMF and SPMF.

user/item representation is crucial to complete the sparse rating matrix especially for the cell values corresponding to the tail items.

We therefore present a sparse probabilistic matrix factorization method (SPMF) by utilizing Laplace distribution to model the item/user factor vector. Laplace distribution [Mohamed *et al.*, 2011] makes the most elements of each factor vector close to zero, which on the one hand is beneficial for SPMF to distinguish the relevant and irrelevant latent features for each user/item. Meanwhile, the tails in Laplace distribution are comparatively heavy, which on the other hand is rewarding to identify the tail items and partially solve the long-tail problem. Because Laplace distribution is non-smooth, the Bayesian inference of SPMF is not analytically tractable. In this paper, we express it as a scale-mixture of Gaussian distribution and exponential density, and employ the Markov chain Monte Carlo technique to perform the Bayesian inference. The second distinguishing feature of our work is the distributed Gibbs sampling algorithm for training the SPMF model, which can efficiently deal with large-scale, sparse and very imbalanced dataset such as 1M MovieLens and 100M Netflix data.

The rest of the paper is organized as follows. The related work is described in Section 2. Section 3 gives the proposed sparse probabilistic matrix factorization model and the distributed Gibbs sampling algorithm for model inference. A series of experimental results on real world datasets are listed and discussed in Section 4. A brief conclusion and future work are shown in Section 5.

2 Related Work

Our proposed model can be taken as a variant of Probabilistic Matrix Factorization. Let us briefly review PMF and Bayesian PMF.

2.1 Probabilistic Matrix Factorization

Probabilistic Matrix Factorization (PMF) is a probabilistic linear model with Gaussian observation noise [Salakhutdinov and Mnih, 2007] . Figure 1(a) shows the graphical model of PMF using the plate convention. The conditional distribution of observed rating data $\boldsymbol{R} \in \Re^{M \times N}$ on the latent user factor $\boldsymbol{U} \in \Re^{D \times M}$ and item factor $\boldsymbol{V} \in \Re^{D \times N}$ is modeled via

$$p(\boldsymbol{R}|\boldsymbol{U}, \boldsymbol{V}, \sigma^2) = \prod_{i=1}^{M} \prod_{j=1}^{N} [\boldsymbol{N}(\boldsymbol{R}_{ij}|\boldsymbol{U}_i^T \boldsymbol{V}_j, \sigma^2)]^{\boldsymbol{I}_{ij}} \quad (1)$$

where M is the number of users, N is the number of items, D is the number of latent features. $\boldsymbol{N}(x|\mu, \sigma^2)$ denotes Gaussian distribution with mean μ and variance σ^2. $\boldsymbol{I}_{ij}$ is an indicator function, $\boldsymbol{I}_{ij} = 1$ if the ith user is associated with the jth item, otherwise $\boldsymbol{I}_{ij} = 0$. The user factor and item factor are assumed to follow Normal distribution, i.e.,

$$p(\boldsymbol{U}|\sigma_U^2) = \prod_{i=1}^{M} \boldsymbol{N}(\boldsymbol{U}_i|0, \sigma_U^2 \boldsymbol{I}) \quad (2)$$

$$p(\boldsymbol{V}|\sigma_V^2) = \prod_{j=1}^{N} \boldsymbol{N}(\boldsymbol{V}_j|0, \sigma_V^2 \boldsymbol{I}) \quad (3)$$

In [Salakhutdinov and Mnih, 2007], maximizing the log-posterior of (1) is implemented by minimizing the sum-of-squares error function with quadratic regularization terms. However, restricting all features to the same regularization level limits the flexibility of the model. Moreover, it is computationally very expensive to search for appropriate values of the regularization parameters.

2.2 Bayesian Probabilistic Matrix Factorization

Bayesian probabilistic matrix factorization (BPMF) [Salakhutdinov and Mnih, 2008] introduces Gaussian-Wishart priors for the hyperparameters, as shown in Figure 1(b). Among them, the prior distributions over $\boldsymbol{U}$ and $\boldsymbol{V}$ are assumed to be Gaussian:

$$p(\boldsymbol{U}|\mu_U, \Lambda_U) = \prod_{i=1}^{M} \boldsymbol{N}(\boldsymbol{U}_i|\mu_U, \Lambda_U^{-1}) \quad (4)$$

$$p(\boldsymbol{V}|\mu_V, \Lambda_V) = \prod_{j=1}^{N} \boldsymbol{N}(\boldsymbol{V}_j|\mu_V, \Lambda_V^{-1}) \tag{5}$$

The hyperparameters $\Theta_U = \{\mu_U, \Lambda_U\}, \Theta_V = \{\mu_V, \Lambda_V\}$ are assumed to be Gaussian-Wishart distribution. By maximizing the log-posterior of the model over both parameters and hyper parameters, BPMF can automatically control the model complexity. Later, Shi et al. [Shi *et al.*, 2013] proposed sparse covariance matrix factorization (SCMF) by imposing Laplace prior on the covariance matrices of $\boldsymbol{U}$ and $\boldsymbol{V}$ to consider the feature correlations and prevent overfitting. Although BPMF and SCMF obtain promising results, they, like PMF, assume that the factors follow Gaussian distribution, which is not justified especially when the data are extremely sparse.

In the next section, we will give a sparse probabilistic matrix factorization model and a distributed inference algorithms based on Gibbs sampling technique.

3 Sparse Probabilistic Matrix Factorization

To achieve sparsity, the sparsity-favoring distributions can be employed which prefer a high excess kurtosis, i.e., a high peak with heavy tails. The set of sparsity-favoring distributions includes spike-and-slab, Student-t, Laplace, and Gamma distribution [Polson and Scott, 2010]. Among them, Laplace distribution is log-concave, leading to a posterior whose log density is a concave function and has a single local maximum, which is essential to design robust and easy-to-use algorithm [Paninski, 2005]. The probability density function (P.D.F.) of Laplace distribution is defined as:

$$\boldsymbol{L}(x|\mu, \rho) = \frac{1}{2\rho} \exp\left(-\frac{|x-\mu|}{\rho}\right). \tag{6}$$

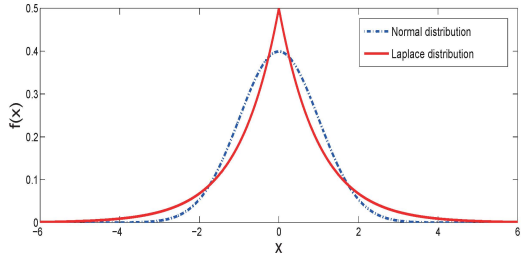


Figure 2: The P.D.F of Gaussian distribution with ($\mu = 0$, $\sigma = 1$) and Laplace distribution with ($\mu = 0$, $\rho = 1$).

As shown in Figure 2, Laplace distribution has higher excess kurtosis than Gaussian distribution at the mean point, which means that the variable x has greater chance to concentrate close to zero. Meanwhile, Laplace distribution has two comparative heavy tails allowing for occasional large values, which is beneficial for model the tail items in MF-based collaborative filtering methods.

3.1 SPMF Model

In SPMF, the conditional distribution of the observed ratings $\mathbf{R}_{ij}$ is still i.i.d. normal distribution with mean $\boldsymbol{U}_i^T \boldsymbol{V}_j$ and variance σ^2, as shown in (1). To get a full Bayesian approach, we introduce an inverse Gamma distribution ($\Gamma^{-1}(\alpha, \beta)$) with shape α and rate β to model the Gaussian noise variance σ^2:

$$p(\sigma^2|\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} (\sigma^2)^{-(\alpha+1)} \exp\left(-\frac{\beta}{\sigma^2}\right) \tag{7}$$

The inverse Gamma distribution is already proved to be effective in modeling the unknown variance of a normal distribution [Witkovsky, 2001].

In order to effectively characterize each user and item, we try to select the most informative latent features to represent them. Thus, we employ Laplace distribution to model the latent user factor $\boldsymbol{U}$ with zero mean and η_U scale, and model the latent item factor $\boldsymbol{V}$ with zero mean and η_V scale.

$$\begin{aligned} p(\boldsymbol{U}|\eta_U) &= \prod_{i=1}^{M} \boldsymbol{L}(\boldsymbol{U}_i|0, \eta_U) \\ p(\boldsymbol{V}|\eta_V) &= \prod_{j=1}^{N} \boldsymbol{L}(\boldsymbol{V}_i|0, \eta_V) \end{aligned} \tag{8}$$

Here, we introduce the Generalized Inverse Gaussian (***GIG***) distribution to model the scales (η_U and η_V) of Laplace distribution for obtaining a fully Bayesian treatment of SPMF and enhancing the model robustness. In [Zhang *et al.*, 2012], it has been demonstrated that Laplace mixture with ***GIG*** is beneficial to define a regularizer for variable selection, which is useful for our SPMF model to select the most informative latent features for each item or user. Let $x = (\eta_U)_{ki}$, then

$$\boldsymbol{GIG}(x|\gamma, a, b) = \frac{(a/b)^{\gamma/2}}{2K_\gamma(\sqrt{ab})} x^{\gamma-1} \exp\Big(-(ax + b/x)/2 \Big) \tag{9}$$

where $K_\gamma(\cdot)$ is a modified Bessel function of the second kind with the index γ. The distribution of $(\eta_V)_{kj}$ has the similar form.

According to the Bayesian theory, the posterior distribution over latent factors $\boldsymbol{U}$ and $\boldsymbol{V}$ can be modeled as

$$\begin{aligned} p(\boldsymbol{U}, \boldsymbol{V}|\boldsymbol{R}, \sigma^2, \eta_U, \eta_V) &= \frac{p(\boldsymbol{R}|\boldsymbol{U}, \boldsymbol{V}, \sigma^2) p(\boldsymbol{U}|\eta_U) p(\boldsymbol{V}|\eta_V)}{p(\boldsymbol{R}|\sigma^2)} \\ &\propto p(\boldsymbol{R}|\boldsymbol{U}, \boldsymbol{V}, \sigma^2) p(\boldsymbol{U}|\eta_U) p(\boldsymbol{V}|\eta_V). \end{aligned} \tag{10}$$

Then, the overall generative process can be summarized as follows.

- Draw scales $(\eta_U)_{ki}$ and $(\eta_V)_{kj}$ from $\boldsymbol{GIG}(\gamma, a, b)$.
- Draw each user factor vector $\mathbf{U}_i$ from $\boldsymbol{L}(0, (\eta_U)_i)$ and each item factor vector $\mathbf{V}_j$ from $\boldsymbol{L}(0, (\eta_V)_j)$.
- Draw σ^2 from $\Gamma^{-1}(\alpha, \beta)$
- Draw each observed rating $\boldsymbol{R}_{ij}$ from $\boldsymbol{N}(\boldsymbol{U}_i^T \boldsymbol{V}_j, \sigma^2)$.

and the graphical model of SPMF is shown in Figure 1(c).

3.2 Inference

Markov chain Monte Carlo (MCMC)-based methods are widely applied to approximate the predictive distribution (like

(10)) [Neal, 1993]. Its key idea is to construct a Markov chain that will evenly converge to the posterior distribution of the model with the given data. Each state of the Markov chain is used as sample of the desired distribution. When the conditional distributions can be sampled easily, Gibbs sampling is the simplest but efficient algorithm. In SPMF model, however, it is not easy to sample from a non-smooth Laplace distribution. Fortunately, Laplace distribution can be equivalently expressed as a scaled mixture of Gaussians [Andrews and Mallows, 1974], i.e., an infinite Gaussian mixture with an exponential distribution like

$$\boldsymbol{L}(x|\mu,\rho) = \int_0^\infty \boldsymbol{N}(x|\mu,\varepsilon)\exp(\varepsilon|\frac{\rho}{2})d\varepsilon. \tag{11}$$

Obviously, all priors on the parameters and hyperparameters in SPMF are conjugate, thus, we can develop an efficient Gibbs sampling algorithm to infer SPMF model. Gibbs sampling sample each variable from its distribution conditional on the current values of all other variables, i.e., it samples each variable by fixing all others.

Sample parameters $\boldsymbol{U}_i$:

For $\boldsymbol{U}_i$, we can extract all terms related to $\boldsymbol{U}_i$ and use Bayes' rule to obtain

$$\begin{aligned} &p\big(\boldsymbol{U}_i|\boldsymbol{R},\boldsymbol{V},\sigma^2,(\eta_{U})_i\big) \propto \\ &\prod_{j=1}^{N}[\boldsymbol{N}(\boldsymbol{R}_{ij}|\boldsymbol{U}_i^T\boldsymbol{V}_j,\sigma^2)]^{\boldsymbol{I}_{ij}}\boldsymbol{L}(\boldsymbol{U}_i|0,(\eta_{U})_i). \end{aligned} \tag{12}$$

In order to incorporate the alternative expression of Laplace distribution, infinite Gaussian distribution as shown in (11), we introduce a vector $(\lambda_{U})_i \in R^D$, where each element $(\lambda_{U})_{ki}$ is a latent variable with exponential prior, i.e.,

$$p\big((\lambda_{U})_{ki}|(\eta_{U})_{ki}\big) = \exp((\lambda_{U})_{ki}|(\eta_{U})_{ki})$$

for the corresponding $\boldsymbol{U}_{ki}$. In this case, we can further express (12) as

$$p(\boldsymbol{U}_i|\boldsymbol{R},\boldsymbol{V},\sigma^2,(\lambda_{U})_i) = \boldsymbol{N}(\boldsymbol{U}_i|\mu_i^*,\Lambda_i^*) \tag{13}$$

where

$$\begin{aligned} \Lambda_i^* &= (\lambda_{U})_i^{-1} + \sum_{j=1}^{N}\frac{[\boldsymbol{V}_j\boldsymbol{V}_j^T]^{\boldsymbol{I}_{ij}}}{\sigma^2} \\ \mu_i^* &= [\Lambda_i^*]^{-1}\left(\sum_{j=1}^{N}\frac{[\boldsymbol{R}_{ij}\boldsymbol{V}_j^T]^{\boldsymbol{I}_{ij}}}{\sigma^2}\right) \end{aligned} \tag{14}$$

Sample hyperparameters $(\lambda_{U})_i$ and $(\eta_{U})_i$:

When introducing the vector λ_{U}, we have $p\big((\lambda_{U})_i|\boldsymbol{U}_i,(\eta_{U})_i\big) \propto p(\boldsymbol{U}_i|(\lambda_{U})_i,(\eta_{U})_i)p((\lambda_{U})_i)$ and each element $(\lambda_{U})_{ki}$ has an exponential prior. According to the property of exponential distribution, $(\lambda_{U})_{ki}^{-1}$ follows an inverse Gaussian distribution (denoted as $\boldsymbol{G}^{-1}$) [Zhang *et al.*, 2012]. Then, we can get the conditional distributions of the hyperparameters $(\lambda_{U})_i$ as follows.

$$p\left(\frac{1}{(\lambda_{U})_i}|\boldsymbol{U}_i,(\eta_{U})_i\right) = \boldsymbol{G}^{-1}\left(\frac{\sqrt{(\eta_{U})_i}}{|\boldsymbol{U}_i|},(\eta_{U})_i\right) \tag{15}$$

For the scale of Laplace distribution, $(\eta_{U})_i$, it follows ***GIG*** distribution as shown in (9), i.e.,

$$p\big((\eta_{U})_i|(\lambda_{U})_i,p,a,b\big) = \boldsymbol{GIG}(\gamma+1,(\lambda_{U})_i+a,b)$$

Since it is inefficient to do sampling directly from ***GIG*** distribution, we convert the posterior distribution to a special case of ***GIG*** distribution by setting $\gamma = -1/2$. In this case, ***GIG*** distribution becomes an inverse Gaussian distribution, thus we can sampling $(\eta_{U})_i$ with

$$p((\eta_{U})_i|(\lambda_{U})_i,\gamma,a,b) = \boldsymbol{G}^{-1}\left(\sqrt{\frac{(\lambda_{U})_i+a}{b}},(\lambda_{U})_i+a\right) \tag{16}$$

Sample parameters σ^2 :

Based on (17), the posterior distribution of σ^2 by fixing other variables can be written as

$$\begin{aligned} p(\sigma^2|\boldsymbol{R},\boldsymbol{U},\boldsymbol{V}) &= \Gamma^{-1}(\alpha_{\sigma^2},\beta_{\sigma^2}) \\ \alpha_{\sigma^2} &= \frac{M\times N}{2}+1+\alpha \\ \beta_{\sigma^2} &= \frac{1}{2}\sum_{ij}(\boldsymbol{R}_{ij}-\boldsymbol{U}_i^T\boldsymbol{V}_j)^2+\beta \end{aligned} \tag{17}$$

The sampling scheme about parameters $\boldsymbol{V}_j$ and hyperparameters $(\lambda_{V})_j$ and $(\eta_{V})_j$ is similar to $\boldsymbol{U}_i$, $(\lambda_{U})_i$ and $(\eta_{U})_i$ respectively.

3.3 Distributed Sampling

Note that the columns of $\boldsymbol{U}$ and $\boldsymbol{V}$ are independent, we can sample them in parallel. More precisely, when sampling $\boldsymbol{U}$, the i-th column of $\boldsymbol{U}$, can be obtained by using only the i-th column of rating matrix $\boldsymbol{R}$, which holds the i-th user's preference, and all the columns of $\boldsymbol{V}$. In such a setting, we can use a parallel broadcast-join [Blanas *et al.*, 2010] to efficiently infer SPMF model.

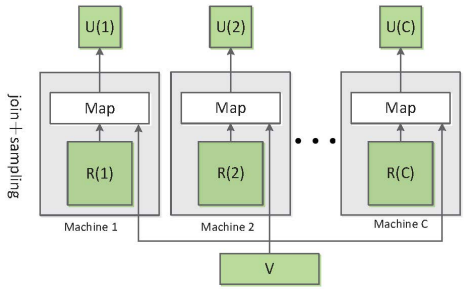


Figure 3: Parallel sampling of factor $\boldsymbol{U}$ by broadcast-join.

As shown in Figure 3,we broadcast the factor $\boldsymbol{V}$ to all participating machines (there are total C machines), which create a hashtable for its contents. The rating data $\boldsymbol{R}$ are stored in the distributed filesystem (DFS) partitioned by its rows (corresponding to users) and forms the input for the map operator. Here $\boldsymbol{R}(1)$ refers to partition 1 of $\boldsymbol{R}$. The map operator reads the i-th row of $\boldsymbol{R}$ and selects all columns of $\boldsymbol{V}$ from the corresponding hashtable. Next, the map operator can sample $\boldsymbol{U}_i$ via (13), $(\lambda_{U})_i$ via (15), $(\eta_{U})_i$ via (16), σ^2 via (17), and write

back its result. This strategy contains multithreaded mappers that leverage all cores of the worker machines for sampling $\boldsymbol{U}$.

The computational complexity of inferring the parameter $\boldsymbol{U}$ and its corresponding hyperparameters is $O(T(\hat{M}D^3 + \hat{C}D^2))$, where T is the number of iterations, D is the number of latent features, $\hat{M}$ and $\hat{C}$ are the maximum number of users and the maximum observed ratings in all machines. The sampling process for $\boldsymbol{V}$ works analogously and its computational complexity is $O(T(\hat{N}D^3 + \hat{C}D^2))$, here $\hat{N}$ is the maximum number of items in all machines. Thus, the total cost of inferring process is $O\big(T(\hat{M}D^3 + \hat{N}D^3 + \hat{C}D^2)\big)$. Obviously, the computational complexity of our algorithm is linearly scalable to the size of rating data matrix, thus, it is practical for handling large-scale dataset.

4 Experimental Results and Discussions

In this section, we compare SPMF with other baseline methods PMF and BPMF using the MovieLens and Netflix datasets.

4.1 Methodology

Datasets: In our experiment, we use MovieLens[1] data with 1M ratings and Netflix[2] data with 100M ratings, where the ratings are ordinal values on the scale 1 to 5. These two datasets are extensively used in the literature to test the performance of recommendation systems. The statistical properties of the datasets are summarized in Table 1. Density indicates the percentage of non-zero cells in rating matrix. Netflix is more challenging than MovieLens. In Netflix, there are over 30% of the items having less than 10 ratings.

Table 1: Summary of Datasets.

Dataset	Users	Items	Ratings	Density
Movielens	6,040	3,952	1M	4.26%
Netflix	480,198	17,770	100M	1.18%

Parameter setting: In PMF, the regularization parameters are tuned from the candidate set $\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10^1, 10^2, 10^3\}$. Following [Salakhutdinov and Mnih, 2008], we set $\mu_0 = 1$, $v_0 = D$, $W_0 = \boldsymbol{I}_{D\times D}$, and variance $\sigma^2 = 1/2$ for BPMF. We employ the code[3] given by the authors .

In the proposed SPMF, we adopt an uninformative prior for the noise variance, i.e., initializing the shape and rate of inverse Gamma distribution via $\alpha = \beta = 0$. Meanwhile, the prior of inverse Gaussian distribution on hyperparameters η_U is initialized by setting $a = 1/D$ and $b = D$, here a is a part of scale in (16) which plays an important role to control the sparsity. Smaller a leads to more sparse factor vector. The reason that we use this setting is that the size of latent features (D) in SPMF may affect the sparsity of latent factor vectors. For example, when D is small, i.e., there are few features, then each latent factor vector may be related to most features, which means that the factor vector will be dense. Otherwise, larger D may lead to sparse factor vector. Thus, we can say D is inverse proportional to the sparsity of factor vector.

Evaluation: The Root Mean Square Error (RMSE) is used to measure the performance of our proposed SPMF and the baselines (PMF and BPMF).

$$RMSE = \sqrt{\frac{1}{|\Omega|}\sum_{(i,j)\in\Omega}(\boldsymbol{R}_{ij} - \widehat{R}_{ij})^2} \qquad (18)$$

where $|\Omega|$ is the number of observed ratings in testing data, $\widehat{R}_{ij}$ is the predicated rating value of the i-th user for the j-th item. Smaller RMSE indicates better recommendation.

4.2 Results and Discussions

In experiments, we randomly select 90% data as training data and the remaining ratings as testing data for ten times, and the average results on testing data are recorded. When selecting training data, every item and user appears at least once.

Effect of latent feature size

We are firstly interested in evaluating the effect of the number of latent features (D) on the proposed SPMF and baselines (PMF and BPMF). Table 2 shows the performance of these models with different latent feature sizes on two datasets.

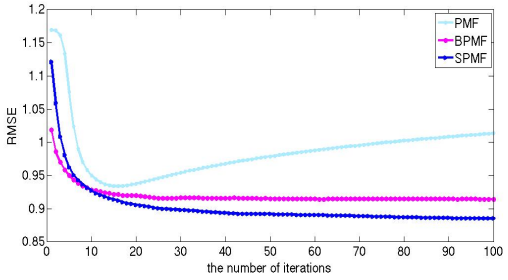


Figure 4: Demonstration of convergence on Netflix data with $D = 50$.

It can be seen that SPMF and BPMF outperforms PMF, which indicates that automatically learning the regularization parameters is beneficial to enhance the model robustness, esp., avoid overfitting problem. With the increase of D, the performances of both SPMF and BPMF steadily improve. To be excited, SPMF consistently performs better than BPMF on all latent feature sizes, and larger D leads to bigger improvement (e.g., SPMF outperforms BPMF by over 1.09% to 1.54% on MovieLens data, 2.73% to 3.46% on Netflix from $D = 10$ to 150). Furthermore, we show the convergence process along iterations (taking Netflix as an example) in Figure 4, which verifies that SPMF like BPMF does not overfit. The factors of PMF are randomly initialized. The factors in BPMF and SPMF are initialized with the MAP estimations obtained by PMF. Since BPMF and PMF assume that the factors follow Gaussian distribution, so that BPMF starts better than SPMF. The main difference between SPMF and BPMF is the prior over the latent factors, which demonstrates that

[1] http://www.grouplens.org/node/73

[2] www.netflixprize.com

[3] http://www.cs.toronto.edu/ rsalakhu/BPMF.html

Table 2: Comparison of RSME under varying latent feature size (D) on MovieLens and Netflix.

	MovieLens			Netflix		
D	PMF	BPMF	SPMF	PMF	BPMF	SPMF
10	0.8788±0.0019	0.8404±0.0024	**0.8312**±0.0021	0.9367±0.0019	0.9208±0.0015	**0.8957**±0.0012
30	0.8605±0.0021	0.8346±0.0021	**0.8256**±0.0025	0.9319±0.0015	0.9156±0.0023	**0.8890**±0.0009
50	0.8855±0.0024	0.8320±0.0022	**0.8233**±0.0022	0.9339±0.0025	0.9143±0.0031	**0.8852**±0.0015
70	0.8896±0.0016	0.8313±0.0024	**0.8207**±0.0008	0.9343±0.0038	0.9133±0.0025	**0.8833**±0.0017
100	0.8896±0.0010	0.8312±0.0007	**0.8192**±0.0011	0.9389±0.0014	0.9128±0.0013	**0.8821**±0.0016
150	0.8894±0.0011	0.8310±0.0010	**0.8182**±0.0012	0.9443±0.0010	0.9125±0.0013	**0.8809**±0.0009

imposing a sparse-favoring distribution on latent factors does bring performance gain especially on sparse Netflix data.

Table 3: Comparison of running time (seconds) under varying latent feature size (D) on MovieLens dataset.

D	10	30	50	70	100	150
BPMF	364	950	2395	4111	7231	13127
SPMF-1	581	1403	2548	7051	11455	19305
SPMF-4	276	618	1017	2950	5018	7435
SPMF-8	225	495	761	2171	3524	5223

Even though larger D can lead to better performance for SPMF and BPMF, it also increases the model complexity because running the Gibbs sampling on more parameters is computationally much more expensive. As listed in Table 3, more running time is needed by both SPMF and BPMF when D grows. All experiments are conducted on the PCs with Intel Core i3-2120 3.3GHz processor and 4 GB RAM. Here SPMF-1 indicates non-distributed sampling. SPMF-4 and SPMF-8 use four and eight machines for distributed sampling respectively. The change of RMSE (less than 0.00001) is taken as the stop condition for both methods. Obviously, distributed sampling in SPMF is helpful to speed up the inferring process.

Long tail recommendation

In modern recommendation systems, only a small part of items are popular, while a large part of items have few rating information. However, such unpopular items, also called as long tail items [Anderson, 2006], often play an important role for recommendation, esp., in commercial world. For example, Amazon successfully makes most of their profit from the long tail products rather than the best selling products.

In order to investigate how our proposed SPMF method deal with the long tail items, we firstly ranking items according to their rating frequency, and then almost evenly separated them into ten groups. Figure 5(a) demonstrates the number of ratings in ten item groups for MovieLens dataset, where the x-axis refers to the boundary of rating frequency in each group. Obviously, they follow long tail distribution. Figure 5(b) and (c) give the average RMSE of each item group for MovieLens and Netflix respectively, which are obtained by recommendation results SPMF, BPMF and PMF with setting $D = 50$. As expected, the proposed SPMF model outperforms two baselines especially for the items in the long tail part.

This result further verifies that SPMF integrating with sparsity-favoring distribution (Laplace distribution) on latent factors has ability to handle sparse rating data. The main reason is that BPMF imposes Gaussian prior on $\boldsymbol{V}$, while SPMF assumes $\boldsymbol{V}$ is generated from Laplace distribution. Such sparsity prior can effectively partition the elements (in each factor vector) into a large set which is close to zero with high probability and a small set which has significant mass on large values. Thus, SPMF can model the long tail items well and significantly enhance the recommendation performance especially for unpopular items.

5 Conclusions and Future Work

In this paper we propose a sparse probabilistic matrix factorization model and a distributed Gibbs sampling algorithm to infer the model. The experimental results have demonstrated that the model can be efficiently trained and successfully applied on sparse and large-scale data by comparing with the state-of-the-art MF-based CF methods. One distinguishing characteristic of our model is that it has ability to deal with long tail items. This makes SPMF have potential commercial value, e.g., it can be used to explore the long tail market and boost the one-stop shopping convenience.

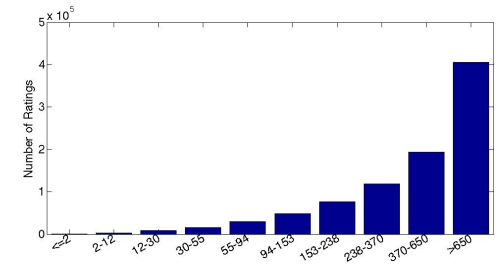
The proposed model benefits from the Laplace distribution for modeling the latent factor, however, we have to empirically tune the size of latent features. It will be nice if we can automatically find the proper D. One more thing is that the Gibbs sampling method for log-linear model has to cost large memory, thus, we will transfer it into robust MF problem [Wang *et al.*, 2012] to reduce the space complexity. Another possible direction is to extend the model and make it able to do cold-start recommending which is a challenging issue [Houlsby *et al.*, 2014]. Last but not least, it is interesting to apply the model in other domains such as e-commerce with large-scale and online data.

Acknowledgments

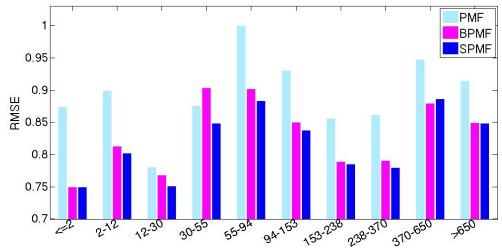
This work was supported in part by the NSFC Grants 61375062 and 61370129, PCSIRT Grant IRT201206 and the Fundamental Research Funds for the Central Universities 2014JBM029 and 2014JBZ005.

References

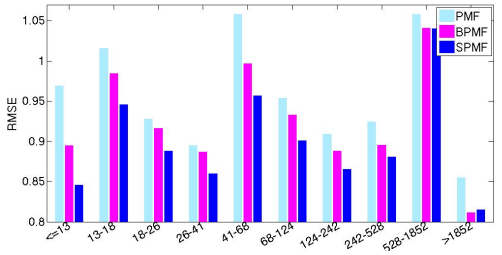
[Adomavicius and Tuzhilin, 2005] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.



(a) Long tail distribution of MovieLens



(b) RMSE on MovieLens



(c) RMSE on Netflix

Figure 5: Comparison of three methods with respect to ten item groups on MovieLens and Netflix.

[Anderson, 2006] Chris Anderson. *The long tail: Why the future of business is selling less of more*. Hyperion, 2006.

[Andrews and Mallows, 1974] David F Andrews and Colin L Mallows. Scale mixtures of normal distributions. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 99–102, 1974.

[Bauer and Nanopoulos, 2014] Josef Bauer and Alexandros Nanopoulos. A framework for matrix factorization based on general distributions. In *Proc. of ACM RecSys*, pages 249–256, 2014.

[Berry *et al.*, 1999] Michael W Berry, Zlatko Drmac, and Elizabeth R Jessup. Matrices, vector spaces, and information retrieval. *SIAM Review*, 41(2):335–362, 1999.

[Blanas *et al.*, 2010] Spyros Blanas, Jignesh Patel, Vuk Ercegovac, and Jun Rao. A comparison of join algorithms for log processing in map reduce. In *Proc. of ACM SIGMOD*. ACM, 2010.

[Chen *et al.*, 2011] Po-Lung Chen, Chen-Tse Tsai, Yao-Nan Chen, Ku-Chun Chou, Chun-Liang Li, Cheng-Hao Tsai, Kuan-Wei Wu, Yu-Cheng Chou, Chung-Yi Li, Wei-Shih Lin, et al. A linear ensemble of individual and blended models for music rating prediction. *KDDCup*, 2011.

[Dror *et al.*, 2012] Gideon Dror, Noam Koenigstein, Yehuda Koren, and Markus Weimer. The yahoo! music dataset and kdd-cup'11. In *KDD Cup*, pages 8–18, 2012.

[Houlsby *et al.*, 2014] Neil Houlsby, Jose Hernandez-Lobato, and Zoubin Ghahramani. Cold-start active learning with robust ordinal matrix factorization. In *Proc. of ICML*, 2014.

[Koren *et al.*, 2009] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 8:30–37, 2009.

[Lakshminarayananw *et al.*, 2011] Balaji Lakshminarayananw, Guillaume Bouchard, and Cedric Archambeau. Robust bayesian matrix factorization. In *Proc. of AISTATS*, pages 425–433, 2011.

[Mohamed *et al.*, 2011] Shakir Mohamed, Katherine Heller, and Zoubin Ghahramani. Bayesian and l1 approaches to sparse unsupervised learning. *arXiv preprint arXiv:1106.1157*, 2011.

[Neal, 1993] Radford Neal. Probabilistic inference using markov chain monte carlo model methods. *Technical Report CRG-TR-93, University of Toronto*, 1993.

[Paninski, 2005] Liam Paninski. Log-concavity results on gaussian process methods for supervised and unsupervised learning. In *Proc. of NIPS*. MIT Press, 2005.

[Polson and Scott, 2010] Nicholas Polson and James Scott. Shrink globally, act locally: sparse bayesian regularization and prediction. *Bayesian Statistics*, 9:1–24, 2010.

[Salakhutdinov and Mnih, 2007] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *Proc. of NIPS*, pages 1257–1264, 2007.

[Salakhutdinov and Mnih, 2008] Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proc. of ICML*, pages 880–887, 2008.

[Shi *et al.*, 2013] Jianping Shi, Naiyan Wang, Yang Xia, Dit Y Yeung, Irwin King, and Jiaya Jia. Scmf: sparse covariance matrix factorization for collaborative filtering. In *Proc. of IJCAI*, pages 2705–2711, 2013.

[Wang *et al.*, 2012] Naiyan Wang, Tiansheng Yao, Jingdong Wang, and Dit-Yan Yeung. A probabilistic approach to robust matrix factorization. In *Proc. of ECCV*, pages 126–139. 2012.

[Witkovsky, 2001] Viktor Witkovsky. Computing the distribution of a linear combination of inverted gamma variables. *Kybernetika*, 37:79–90, 2001.

[Zhang *et al.*, 2012] Zhihua Zhang, Shusen Wang, Dehua Liu, and Michael I Jordan. Ep-gig priors and applications in bayesian sparse learning. *Journal of Machine Learning Research*, 13(1):2031–2061, 2012.