The Gaussian-Gamma Collaborative Filtering: A Hierarchical Bayesian Model For Recommender Systems

Cheng Luo

School of Electronics and Information Engineering, Tongji University, Shanghai, China

Bo Zhang

College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, Shanghai, China

Yang Xiang[1]

School of Electronics and Information Engineering, Tongji University, Shanghai, China

Man Qi

Department of Computing, Canterbury Christ Church University, Canterbury, CT1 1QU, UK

Abstract

The traditional Collaborative Filtering (CF) based recommender system suffers from two key challenges, namely, the normal assumption is not appropriate, and the penalty terms added on the latent feature vectors are difficult to set in advance. Hence we propose a hierarchical Bayesian model based CF and the related inference algorithm. Specifically, we impose a Gaussian-Gamma assumption on the ratings and the feature vectors, and propose a Gibbs sampler for the inference. We show this procedure is meaningful by giving a statistical explanation of the inference. We show the performance of the model by experiments with synthetic datasets and real datasets.

Keywords: template Gaussian-Gamma distribution, recommender system, hierarchical Bayesian model, Gibbs Sampling, performance evaluation

1. Introduction

The Collaborative Filter (CF) has been widely used in recommender systems. It aims at learning the latent features of both the users' and the items' at the same time. When a rating matrix is given, the task of CF is to learn the latent features with the observed ratings and predict the unobserved ratings by these features. It assumes the rating of item j given by user i is a simple inner product of the corresponding latent features. Formally, let U_i denote the latent features of user i and V_j denote the latent features of item j, then the rating $R_{i,j}$, i.e. the (i,j)-th value of the rating matrix R, is $R_{i,j} = U_i' V_j$. This method was first proposed by [1] in the form of an optimization problem.

[1]Corresponding author. Email address:`shxiangyang@tongji.edu.cn`

The statistical explanation of CF was proposed by [2]. It proved the ratings of the dataset could be seen as normally distributed random variables. That is, the rating $R_{i,j}$ is a normal random variable with mean $U_i'V_j$ and a fixed variance which is set empirically. However, the normal distribution is sensitive to outliers. It is widely accepted that when a dataset is contaminated by noise, its mean and standard deviation can be largely affected because the normal distribution has a light tail. Furthermore, the fixed variance in a model is also problematic. Since the experiences of users have a large diversity, the variances of the ratings should be different from one to another. Consequently, fixing a common variance for all ratings is inappropriate. With all these facts, the normal distribution is not the best assumption for the random variables of the ratings.

Another challenge posed to the recommender systems is that the model tends to be over-fitting if the latent features do not have any constraint. This problem arises in those machine learning systems where the model is much too flexible than it needs to be. As a result, the training error can be low while the test error remains very high. An effective solution is to add some penalty terms to the variables to confine the model's flexibility. Surprisingly, this simple method exhibits good performance in reducing over-fitting. However, deciding how far to regularize the flexibility is important. Because when the penalty term is set to be a large value exceeding the demand of the model, both the training error and the test error will rise. Hence it is important to choose appropriate penalty terms. For traditional learning methods, these terms are chosen manually through exhaustive training and comparing.

Fortunately, the recent development of probabilistic models shed some light on the above problems. Probability is a powerful tool to deal with uncertain data. Hence it has been widely used in machine learning [3, 4, 5], data mining [3, 6] and other relevant areas [7, 8, 9, 10, 11, 12]. Particularly, we use the hierarchical Bayesian model to solve the mentioned problems. Instead of a Gaussian distribution for $R_{i,j}$, we substitute it with a Gaussian-Gamma distribution, i.e. we add a Gamma distribution for the precision (the inverse of the variance) and form a higher hierarchy for the model (see Figure 1). For the penalty terms we also add higher distributions for them and hence the distribution of the latent features U_i, V_j is distributed as Gaussian-Gamma distribution as well. The Gaussian-Gamma distribution can be seen as an extension of the famous $\mathcal{T}$ distribution, which is a more robust distribution than the Gaussian distribution. It has a heavier tail and a larger kurtosis while the degree of freedom is small, which signifies the mean of the distribution will be less affected by outliers than the Gaussian distribution. Therefore $\mathcal{T}$, or the extension Gaussian-Gamma distribution, is a better choice for recommender systems.

The contribution of our work includes the following. Firstly, we construct a hierarchical Bayesian model for recommender systems, which assumes that the ratings and the latent features are distributed as Gaussian-Gamma distribution. We call this model the Gaussian-Gamma CF, or GGCF. Secondly, we present an inference algorithm for GGCF which is already tested by a multi-stage Gibbs sampling [13] and proves to be very meaningful. In experiments, we test the model using a synthetic dataset as well as real datasets to show that the Gaussian-Gamma assumption indeed grants a heavier tail and as a result, the prediction errors become lower.

The rest of the paper is organized as follows. Section 2 introduces the related work including the traditional CF and the regularized CF in a probabilistic perspective of view. In Section 3, we give the description of GGCF, the inference procedure, and the way to predict the missing ratings. In Section 4, we show experiments with the synthetic dataset, the MovieLens dataset, and the Book crossing dataset. In Section 5, we give an analytic analysis of the proposed model. In Section 6, we discuss the regularizing terms in machine learning language. Finally, we conclude the paper in Section 7.

2. Related work

For a rating matrix R, CF attempts to find U_i and V_j which minimize the loss function $\|R_{i,j} - U_i'V_j\|^2$ for all the observed $R_{i,j}$. To prevent over-fitting, [1] added some penalty terms and modified the loss function to $L = \sum_{i,j} \|R_{i,j} - U_i'V_j\| + \sum_i \lambda_{1,i}\|U_i\| + \sum_j \lambda_{2,j}\|V_j\|$, where $\lambda_{1,i}$ and $\lambda_{2,j}$ are regularizing weights. Since the values $\lambda_{1,i}$ and $\lambda_{2,j}$ are difficult to set, [14] used another way to regularize the latent features. They added upper and lower bounds on the estimated missing ratings. Low rank representation was also introduced to the recommender systems. Srebro *et al.* [15] proposed using low rank representation to regularize the collaborate filtering. Add social relations of the users are also invited to regularize the model, see [16] and [17] for reference.

Besides the above example of RCFs, probabilistic explanations are introduced to the traditional CF. Mnih *et al.* [2] imposed a normal assumption on the ratings, which states that the rating $R_{i,j}$s are independent and are normally distributed with mean $U_i'V_j$ and a fixed variance. This probabilistic model is equivalent to the traditional CF. Since the maximum of the log-likelihood function of this model is equal to minimize the loss function $\|R_{i,j} - U_i'V_j\|$. Moreover, if normal assumptions are imposed on the feature vectors U_i and V_j, the log-likelihood function of this model is equal to the traditional loss function with penalty terms.

However, few works addressed the appropriation of the normal assumption which lies at the foundation of the model. Meanwhile, hierarchical Bayesian models have been introduced to machine learning community for a long time, and the properties of hierarchical Bayesian models have been deeply analyzed [18, 19]. One successful instance of these models is the Bayesian analysis of the regression models [20, 21, 22, 23, 24]. Instead of normal assumptions, these models assume a $\mathcal{T}$ distribution, or, a Gaussian-gamma distribution on the response variables. Since the $\mathcal{T}$ distribution has a complex density function, we always split it into a Gaussian-Gamma distribution and use sampling methods in the inference. The properties of the Gaussian-Gamma distribution is left in the next section.

Stochastic Gradient Descent (SGD) [25] is one of the most popular methods in learning the parameters of CF. This is because SGD is straightforward and effective. Paterek *et al.* [26] improved the performance of SGD by combining matrix factorization with baseline estimate. Alternating Least Squire (ALS) [27] is another useful algorithm to compute the feature vectors, and its advantage is that the algorithm can be implemented on a parallel platform. However, these algorithms are not applicable in a hierarchical Bayesian model. The reason is the parameters in the latter are considered to be random variables and a full posterior analysis is required. However, we can use Gibbs sampling [28, 13] to approximate the posterior of the random variables. When all the conditional probabilities have closed form, it can be implemented immediately. As will be shown in the next section, we can give the conditional probabilities for all the involved random variables.

3. The Hierarchical Collaborative Filter

3.1. Properties of Gaussian-Gamma distribution

The Gaussian distribution is widely used in statistics and machine learning applications. It contains the following properties. Firstly, Gaussian distribution is ubiquitous because of the central limit theorem and the law of large numbers. Secondly, it has a symmetric density and thus its mean equals its median. Thirdly, the Gaussian distribution belongs to the exponential family. Hence it has conjugate prior and the conditional probabilities for the parameters are tractable. Lastly, it has a light tail. The first three properties are the main reasons why it is applied in most of the models while the last property confines its usage. Because a light tail means the model is sensitive to outliers.

The $\mathcal{T}$ distribution resembles the Gaussian distribution in some way but a heavy tail. We use Kurtosis to test the distributions whether a distribution has heavy tails or not. If the Kurtosis of a distribution is greater than 3, it is called a heavy tail distribution. On the contrary, when the Kurtosis is equivalent to or smaller than 3, it is said to have a light tail. The Kurtosis of a Gaussian distribution is exactly 3, and thus it is not a heavy tailed distribution.

The degree of freedom controls the Kurtosis of a $\mathcal{T}$ distribution. When the degree of freedom is small, the Kurtosis of $\mathcal{T}$ is large, and when the degree of freedom is large, the Kurtosis is small. Particularly, when the degree of freedom is greater than 30, the density of the $\mathcal{T}$ distribution almost coincides with the curve of some Gaussian distribution. The $\mathcal{T}$ distribution is also symmetrically formed and it can be seen as an infinite mixture of the Gaussian distributions. That means if X is a Gaussian distributed random variable with mean μ and precision λ, and if λ is a Gamma distributed random variable with parameters $(\nu/2, \nu/2)$, where $\nu > 0$. Then the marginal density of X integrating out the precision λ coincides with the $\mathcal{T}$ distribution with mean μ and degree of freedom ν. Formally, the density of a $\mathcal{T}$ distribution with mean μ and degree of freedom ν coincides with

$$\int \mathcal{N}(X|\mu, 1/\lambda)\mathcal{G}(\lambda|\nu/2, \nu/2)d\lambda.$$

However, one of the disadvantages of the $\mathcal{T}$ distribution is that it is not in the exponential family. Hence its posterior does not have an analytical form whatever prior is added. To overcome this problem, we substitute the $\mathcal{T}$ distribution with a Gaussian-Gamma distribution, which means we place a Gaussian distribution for the observations, and further put a Gamma distribution for the precision of the Gaussian. This substitution makes all the involved random variables have analytical conditional density. That is because Gaussian and Gamma distributions both belong to the exponential family. In order to make the model more suitable for different occasions, we can further relax the parameters of the Gamma distribution to have different values.

Gaussian-Gamma distribution has been successfully used in the applications when a robust or a sparse model is required. Typical applications include the regression analysis [20, 22, 24] and the Independent Component Analysis (ICA) [21]. These models have similar scenarios with our application. In the regression models, Gaussian-Gamma distribution is utilized instead of a Gaussian prior when the residual is not normally distributed. And in ICA, it is necessary to select a prior different from Gaussian. Because this is the only way we can recover the mixing matrix, where a Gaussian-Gamma distribution works well [21].

3.2. Model description

Let $R \in \mathbb{R}^{M\times N}$ denote a rating matrix with M users rating for N items, where its element $R_{i,j}$ is the rating given to item j by user i. Let matrices $U \in \mathbb{R}^{K\times M}$ and $V \in \mathbb{R}^{K\times N}$, respectively, characterize the users and the items. We model and formalize the relationship between R and U, V as follows

$$\begin{aligned} &R_{i,j} \sim \mathcal{N}(U_i'V_j, \lambda_{i,j}^{(-1)}), \\ &\lambda_{i,j} \sim \mathcal{G}(a,b), \quad i = 1, ..., M, \quad j = 1, ..., N, \end{aligned} \tag{1}$$

where $\mathcal{N}$ and $\mathcal{G}$ denote the normal distribution and Gamma distribution respectively; while $U_i'V_j$ is the inner product of the vectors U_i and V_j which are the i-th column of matrix U and j-th column of matrix V, and $\lambda_{i,j}$ is the reciprocal of variance and is known as the *precision*. The parameters a_λ and b_λ represent the shape and rate of a Gamma distribution respectively. The relationship of these variables are shown in Figure 1.

In fact, this normal-Gamma prior of the ratings is technically equal to a $\mathcal{T}$ prior. By integrating out the precision $\lambda_{i,j}$, we can easily obtain the density of $R_{i,j}$ as

$$\begin{aligned} &\int_{-\infty}^{\infty} \mathcal{N}(R_{i,j}|U_i'V_j, \lambda_{i,j}) \cdot \mathcal{G}(\lambda_{i,j}|a_\lambda, b_\lambda) d\lambda \\ &= \frac{b_\lambda^{a_\lambda}\Gamma(1/2 + a_\lambda)}{\Gamma(a_\lambda)\sqrt{2\pi}} \left(\frac{1}{2}(R_{i,j} - U_i'V_j)^2 + b_\lambda\right)^{-(1/2+a_\lambda)}. \end{aligned} \tag{2}$$

If we set $a_\lambda = b_\lambda$, the term on the right side of equation 2 is just the density of the $\mathcal{T}$ distribution with freedom $2a_\lambda$ and mean $U_i'V_j$.

Hence the ratings are independently distributed as a Gaussian-Gamma distribution which enjoys the heavy tail property. It is known to us that most of the probabilities are held in the interval centered at the expectation with radius 3 times the standard deviation for a Gaussian distribution. Hence if there are any observation far from the mean, say, distant from the mean greater than 3 times the standard deviation, then the mean of the Gaussian distribution will be strongly changed just because of these outliers. This means the Gaussian distribution is sensitive to outliers. However, In a recommender system, most of the ratings are missing in real datasets, and some of the ratings are given casually, outliers exist for most of the cases. Furthermore, since the variances of the ratings for the users change from one to another, it is obviously inappropriate to fix a common variance for the whole data set. These problems are solved for the Gaussian-Gamma distribution obviously. Firstly, it is a heavy tail distribution, and hence it is robust to outliers. Secondly, as shown in equation (1), there is a precision $\lambda_{i,j}$ associated with every rating $R_{i,j}$, the diversity for the users is achieved.

This hierarchical structure also allows a simple two-stage Gibbs sampler for the sampling of the Gaussian-Gamma distribution. The implementation is the iteration of the following two steps repeatedly.

- Sample λ from $\mathcal{G}(a, b)$, and
- Sample x from $\mathcal{N}(\mu, \lambda^{-1})$,

where μ represents the mean of the $\mathcal{T}$ distribution.

Besides the user-item ratings, the latent features U_i and V_j are assumed to be distributed as Gaussian-Gamma distributions as well. The conditional distributions are shown as

$$\begin{aligned} U_i &\sim \mathcal{N}(0, \lambda_{U_i}^{-1} I), \\ \lambda_{U_i} &\sim \mathcal{G}(a_U, b_U), \quad i = 1, ..., M, \\ V_j &\sim \mathcal{N}(0, \lambda_{V_j}^{-1} I), \\ \lambda_{V_j} &\sim \mathcal{G}(a_V, b_V), \quad j = 1, ..., N, \end{aligned} \tag{3}$$

where $I \in \mathbb{R}^{K \times K}$ is an identity matrix. The precisions λ_{U_i} and λ_{V_j} in GGCF play the role of penalty term in RCF, and they are used to prevent the model from over-fitting. However, in RCF, the values of these parameters are set empirically, and it requires a heavy time overhead and needs several times of trying and comparing to set. Here, we impose a simple prior on these parameters so that they can be set automatically in the training phase of the model.

3.3. Inference

The Gibbs sampling is applied in GGCF as the basic tool in inference. It requires the analytical form of the conditional distributions for all the parameters involved. In the following, we need symbols S_i denoting the set of the items rated by user i and S'_j denoting the set of the users who give ratings to item j. Generally speaking, we need the following conditional distribution $p(\lambda_{i,j}|a, b, R_{i,j})$, $p(U_i|R_{i,j}, \lambda_{U_i})$, $p(V_j|R_{i,j}, \lambda_{V_j})$, $p(\lambda_{U_i}|U_i, a_U, b_U)$, $p(\lambda_{V_j}|V_j, a_V, b_V)$.

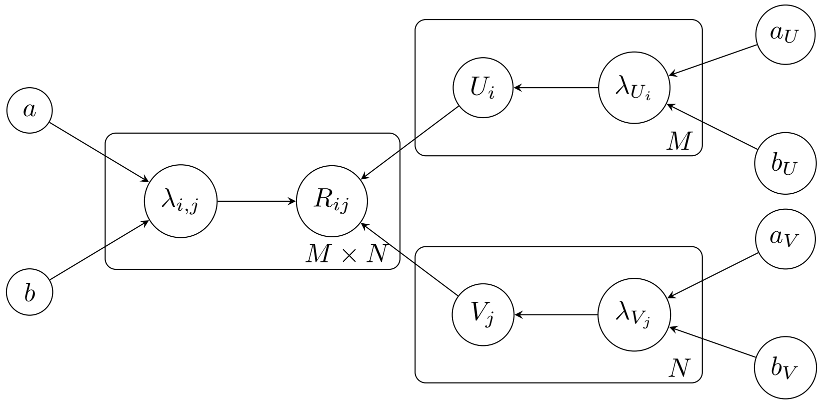


Figure 1: The proposed Bayesian CF model

155 We summarize the conditional distributions here and explain the meaning of them in the following sequel.

$$\begin{aligned}
\lambda_{i,j} &\sim \mathcal{G}(a+1/2, b+(R_{i,j}-U_i'V_j)^2/2),\\
U_i &\sim \mathcal{N}(\tilde{\mu}_{U_i}, \tilde{\Sigma}_{U_i}),\\
V_j &\sim \mathcal{N}(\tilde{\mu}_{V_j}, \tilde{\Sigma}_{V_j}),\\
\lambda_{U_i} &\sim \mathcal{G}(a_U+1/2, b_U+U_i'U_i/2),\\
\lambda_{V_j} &\sim \mathcal{G}(a_V+1/2, b_V+V_j'V_j/2),
\end{aligned} \tag{4}$$

where

$$\begin{aligned}
\tilde{\Sigma}_{U_i} &= \left(\sum_{j\in S_i}\lambda_{i,j}V_jV_j' + \lambda_{U_i}I\right)^{-1},\\
\tilde{\mu}_{U_i} &= \tilde{\Sigma}_{U_i}\sum_{j\in S_i}\lambda_{i,j}R_{i,j}V_j,\\
\tilde{\Sigma}_{V_j} &= \left(\sum_{i\in S_j'}\lambda_{i,j}U_iU_i' + \lambda_{V_j}I\right)^{-1},\\
\tilde{\mu}_{V_j} &= \tilde{\Sigma}_{V_j}\sum_{i\in S_j'}\lambda_{i,j}R_{i,j}U_i.
\end{aligned} \tag{5}$$

The above conditional distributions can describe the real situation well. In fact, the conditional distributions of the latent features U_i and V_j have strong connections with the regularized least square solutions of the linear equations

$$F_iU_i = r_i,$$

where

$$r_i = \begin{bmatrix} R_{i,j_1} \\ R_{i,j_2} \\ \vdots \\ R_{i,j_{|S_i|}} \end{bmatrix}, \quad F_i \ = \left[V_{j_1}, \cdots, V_{j_{|S_i|}} \right].$$

The above linear equation means when the latent features of the users and the items are given, the rating $R_{i,j}$ is an inner product of the latent features of the corresponding user and item. This is just the assumption of the collaborative filtering. However, because of the sparsity of the rating matrix, this linear equation may not have analytical solutions, hence the least square solution is required. Moreover, the regularized terms in the least square method are used to guarantee that the least square solution exists uniquely. Since the least square solution requires a computation of the inverse of a matrix, we need to make sure it is invertible. Hence the regularized terms are inserted. Formally, let $Ax = b$ be a linear equation then it has the least square solution

$$\hat{x} = (A'A)^{-1}A'b.$$

However, since the matrix $A'A$ is not always non-singular, we usually add a fixing term ϵI. Therefore the solution is changed to

$$\hat{x} = (A'A + \epsilon I)^{-1}A'b.$$

This kind of fixing is originally proposed by [29] and ϵ is called the regularity term. The intuition of this fixing method is motivated by the fact that any diagonal dominated matrix is non-singular, and thus we can insert ϵI in with ϵ large enough so the fixed matrix is non-singular.

From equation (5) we know that when we update U_i, F_i plays the role of A, r_i plays the role of b and the regularity term is

$$\sqrt{\Lambda}_i = \begin{bmatrix} \sqrt{\lambda}_{j_1} & & & \\ & \sqrt{\lambda}_{j_2} & & \\ & & \ddots & \\ & & & \sqrt{\lambda}_{j_{|S_i|}} \end{bmatrix},$$

which is a diagonal matrix constituted by the square root of the precisions. The update of V_j has a similar explanation. We only need to switch the role of U and V, and form the linear equation like

$$F'_j V_j = r'_j,$$

where

$$r'_j = \begin{bmatrix} R_{i_1,j} \\ R_{i_2,j} \\ \vdots \\ R_{i_{|S'_j|},j} \end{bmatrix}, \quad F'_i \ = \left[U_{i_1}, \cdots, U_{i_{|S'_j|}} \right].$$

Correspondingly, the regularity terms are constituted by the precisions of the user latent features

$$\sqrt{\Lambda}_j = \begin{bmatrix} \sqrt{\lambda}_{i_1} & & & \\ & \sqrt{\lambda}_{i_2} & & \\ & & \ddots & \\ & & & \sqrt{\lambda}_{i_{|S_j|}} \end{bmatrix}.$$

The update of λ_{U_i} in equation (4) shows that the norm of U_i has an inverse relationship with the precision λ_{U_i}. For a Gamma distributed random variable, its expectation is the quotient of the shape over the rate. Hence the expectation of λ_{U_i} is $(a_U + 1/2)/(b_U + U_i'U_i/2)$. This effect is the same of the regularity terms in the traditional RCF. For this model, when the regularity is large, the norm of the parameters will be small and when the regularity is small, the norm of the parameters is large. Similarly, we can give the explanation of the update of λ_{V_j}. The meaning of the update of $\lambda_{i,j}$ is quite obvious. By the property of the Gamma distribution, the precision $\lambda_{i,j}$ is inversely proportional to the error $(R_{i,j} - U_i'V_j)^2$, and this is just the meaning of the word "precision".

3.4. The estimation of the parameters in GGCF

In the model of GGCF, it is essential to set the parameters of the Gamma distribution $a_\lambda, b_\lambda, \lambda_U$ and λ_V in advance. A traditional empirical method of setting is to set them to be small positive numbers such as 0.01, because a smaller degree of freedom in a $\mathcal{T}$ distribution leads to a heavier tail and a more robust model. On the contrary, when the freedom approaches 30, the density is approximately normal. In fact, the $\mathcal{T}$ distribution converges to a normal distribution is almost definitely when the freedom goes to be infinite. Taking this into account, we here set these parameters to be a small number like 0.01 to encourage the priors of $R_{i,j}, U, V$ to be a more robust one. From a more straight way, the mean of a Gamma distribution is a/b and the variance is a/b^2, therefore, a small number like 0.01 will keep the mean of the precisions to be 1 and a large variance 100 and thus the precisions can be chosen from a broad range.

There also exists methods that choose the parameters automatically. For instance, the empirical Bayes parameter estimator and the hierarchical Bayesian estimator. Since the empirical Bayes estimator involves a lot of complicated computations, we adopt the hierarchical Bayesian estimator. We further put a prior on the Gamma distribution, and then sample these parameters from the posterior distribution.

Assume that the prior for a_λ, b_λ is $\pi(a_\lambda, b_\lambda)$, and the estimation is got by a two-stage gibbs sampler. Note that for given a_λ, we have b_λ as a Gamma distribution with shape $a_\lambda L$ and rate $\sum_{i,j} \lambda_{i,j}$, where L is the number of ratings. The only difficulty lies in the sampling of a_λ from posterior. Conditional on the prior $\pi(a_\lambda, b_\lambda)$, and all other variables, the full conditional of a_λ is

$$\pi(a_\lambda, b_\lambda) \prod_{i,j} \frac{b_\lambda^{a_\lambda}}{\Gamma(a_\lambda)} \lambda_{i,j}^{a_\lambda - 1} \exp(-b_\lambda \lambda_{i,j}).$$

The sampling of a_λ requires a Metropolis-Hastings random walk. For this specific situation, we first sample a proposal a'_λ from $\mathcal{U}(a_\lambda - \epsilon, a_\lambda + \epsilon)$, where $\mathcal{U}$ denotes the uniform distribution and ϵ is a small positive number. Then accept this proposal a'_λ with probability

$$\rho = \min\left\{ \frac{\pi(a'_\lambda, b_\lambda)}{\pi(a_\lambda, b_\lambda)} \cdot \frac{b_\lambda^{a'_\lambda L} (\prod_{i,j} \lambda_{i,j})^{a'_\lambda} \Gamma(a_\lambda)}{b_\lambda^{a_\lambda L} (\prod_{i,j} \lambda_{i,j})^{a_\lambda} \Gamma(a'_\lambda)}, \quad 1 \right\}.$$

It can be proved that the accepted a'_λ has the same distribution with the required one. The tuning parameter ϵ is chosen to set the average acceptance rate at $50\% - 70\%$. Similar procedures can be implemented for a_U, b_U, a_V, b_V.

3.5. Prediction

Suppose $y = f(x|\theta)$ is the likelihood function with predictor x and the parameter θ, and if $\pi(\theta)$ is the density function of θ, then the predicted value $\hat{y}$ of y is

$$\hat{y} = \int f(x|\theta)\pi(\theta)d\theta.$$

If we have a sample $\{\theta_1, ..., \theta_n\}$ of θ, the prediction can be approximately as

$$\hat{y} = \frac{1}{n}\sum_{i=1}^{n} f(x|\theta_i),$$

by law of large number. In this way, the predicted rating $\hat{R}_{i,j}$ is approximated as

$$\hat{R}_{i,j} = \frac{1}{T}\sum_{t=1}^{T} U'_{i,t} V_{j,t}, \tag{6}$$

where T is the size of the sample generated by the Gibbs sampling.

4. Experiments

4.1. Synthetic data

To verify the performance of our proposed model, we generate a matrix $R \in \mathbb{R}^{100\times100}$ in MATLAB as follows: First we generate two matrices $X, Y \in \mathbb{R}^{100\times5}$ with values from a normal distribution with mean 0 and standard deviation 3, then construct $R = XY'$. Hence R is a matrix with dimension 5. Then we add noise to the elements of the matrix and randomly hide 80% of the them as test data. The remaining values are selected as training data. We repeat the experiments for 5 times with different noises. The mean of the noises are set to 0 but the standard deviations are different, which are 0.01, 0.25, 0.5, 0.75, 1. We aim at showing that our model GGCF can catch it, however the standard deviation changes. The predictor U_i and V_j are all set as with dimension 5.

Table 1: The means of the residuals detected by the model.

deviation	0.01	0.25	0.5	0.75	1
round 1	2.2078e-4	-1.5191e-4	-1.7415e-4	-2.4138e-4	-2.9674e-4
round 2	-8.3260e-4	0.0010	-0.0011	-2.4846e-4	-0.0021
round 3	0.0294	0.0326	0.0303	0.0271	0.0280
round 4	0.0458	0.0425	0.0345	0.0398	0.0428
round 5	0.0436	0.0384	0.0351	0.0355	0.0306

Table 2: The kurtosis of residuals of rsvd and bsvd.

deviation	0.01	0.25	0.5	0.75	1
RCF	3.2945	2.9878	3.5490	3.1095	3.2671
GGCF	4.8770	5.0173	5.0251	4.7892	4.9045

The residual of a model is the difference between the real and the predicted ratings. In our experiment, let $R_{i,j}$ denote the rating given by the user i to the movie j and $\hat{R}_{i,j} = U_i'V_j$ represent the corresponding predicted rating, the residual for this specific rating is

$$err_{i,j} = R_{i,j} - \hat{R}_{i,j}. \tag{7}$$

We show the means of the residuals of all different noises in Table 1, in which it can be seen that all of them are around 0. For every deviation, the experiment is repeated 5 times to rule out opportunity. In Figure 2 we show the standard error of the residual goes hand in hand with the standard deviation of the noise. The residual of GGCF (Figure 3(a)) and RCF (Figure. 3(b)) are presented to show that the different effects of the these model assumptions.

To demonstrate that the GGCF provides stronger robustness than RCF, we would show that the kurtosis of the residual of GGCF is greater than that of RCF. Assume that the kurtosis of a normal distribution is 3, in Table 2 we show that the kurtosis of RCF is around 3 while the kurtosis of GGCF is significantly greater than 3, and it is around 5.

4.2. MovieLens data set

To test the accuracy of the prediction ratings of GGCF, we use the MovieLens data set (www.movielens.org) for training and testing. The MovieLens data set contains about 1 million ratings from 6,040 users for 3,952 movies with about 95% sparsity. We randomly select 100,000 ratings with 943 users for 1682 movies as our data set and split it into a training set of size 80,000 and a test set of size 20,000. The number of ratings given by different users diverse in a large range from 4 to 685, and similarly the number of ratings got by

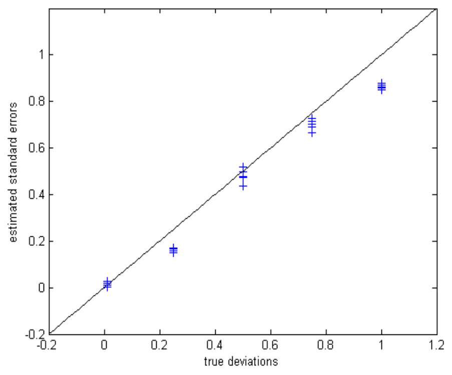
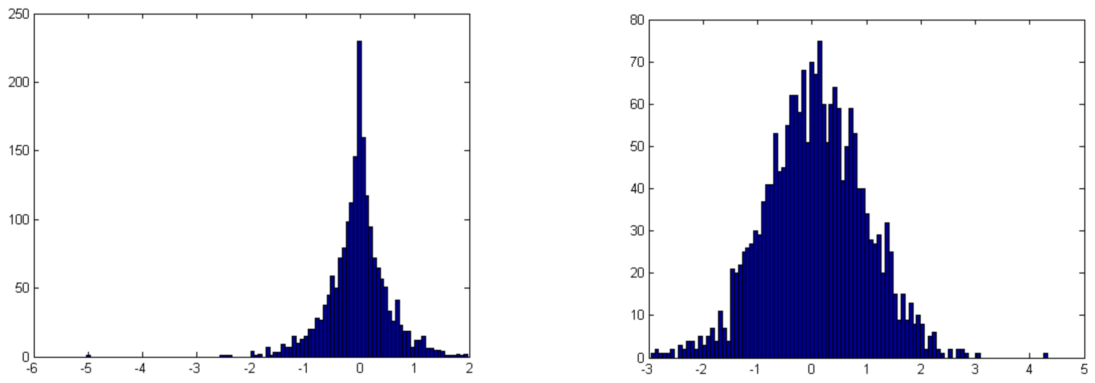


Figure 2: The captured standard errors by GGCF. The blue plus signs represent the standard errors of the residual for different levels of noise. The true standard deviations of the noise imposed on the data set are on the horizontal axis, and the standard errors of the residual are on the vertical axis.



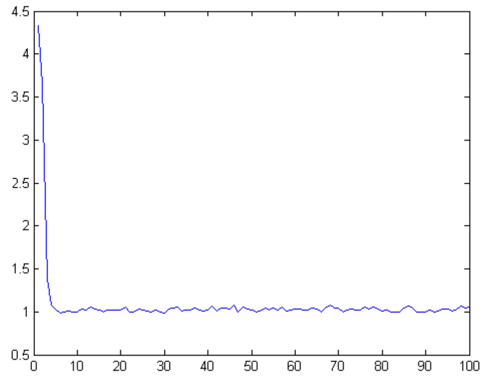
(a) The histogram of the residual of GGCF

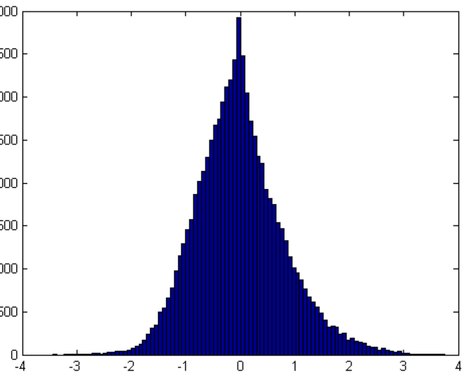
(b) The histogram of the residual of RCF

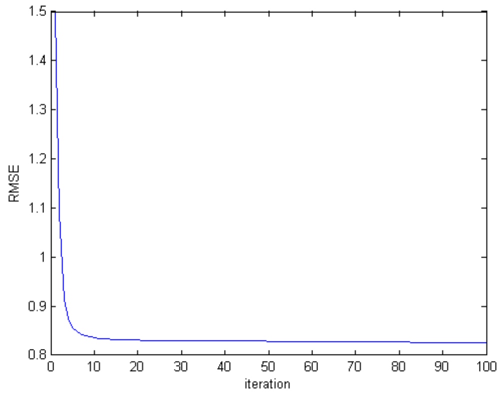
Figure 3: The histograms of the residual of the synthetic data. (a): The histogram of the residuals of GGCF, its tail is heaver than that of RCF. (b): The histogram of the residuals of RCF, it looks like a normal distribution.

different movies also has a large range which is from 1 to 484. The above information means that the penalty given to different feature vectors U_i and V_j should be different.

To train the model of GGCF, we set the parameters $a_\lambda = b_\lambda = a_U = b_U = a_V = b_V = 0.01$ for simplicity, and iterate the Gibbs sampling 100 times. We test the data set by using dimension 2, 3, 5, 10 and 30. In Figure 4, the plots are the results when we set the dimension to be 5 because all the other dimensions produce similar figures. Note that we ignore the plots for all the other dimensions, in that they all produce similar figures. As shown in Figure 4(a), the algorithm of GGCF converges in 5 iterations, which is pretty fast. Figure 4(b) shows the histogram of the residual for the training set. This figure shows that the residual is centered at 0 and has a range $(-3, 3)$. It is worth mentioning that the RMSE in Figure 4(a) seems to


(a) The RMSE of GGCF in every iteration


(b) The histogram of the residual of GGCF


(c) The RMSE of RCF in every iteration

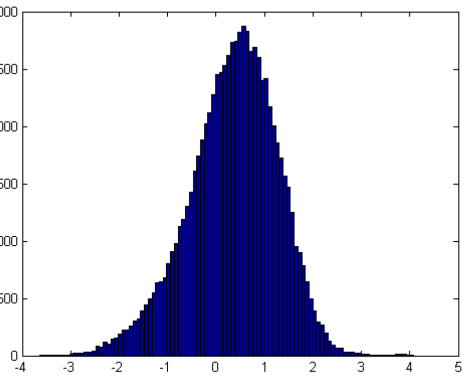
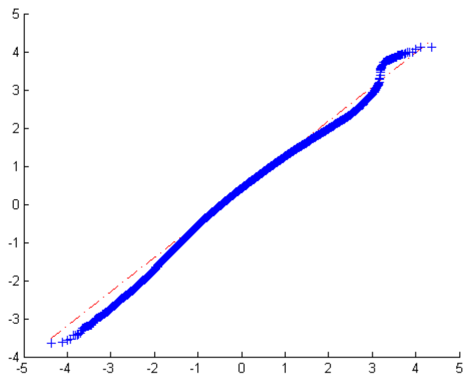

(d) The histogram of the residual of RCF

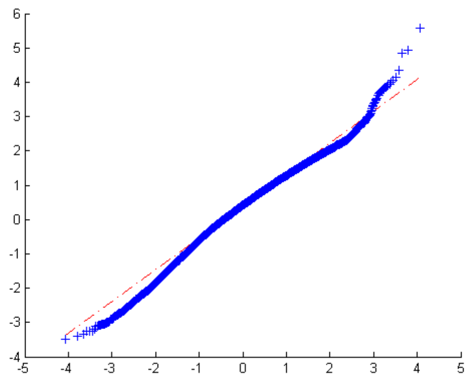
Figure 4: The results of MovieLens data set. (a): RMSE for the training set of GGCF in each iteration, this figure shows the algorithm converges within 5 iterations. (b): The histogram of the residual of GGCF. (c): RMSE for the training set of RCF in each iteration, this figure shows the convergence rate is similar with GGCF. (d): The histogram of the residual of RCF.

be above 1 but the actual RMSE of the this model is indeed below 1. The reason is that equation 6 works like the ensemble of different models and it will rule out some of the unreasonable predictions. For instance, some of the predicted ratings given by only one sampled feature vectors $U'_{i,t}V_{j,t}$ might be greater than 5 or smaller than -5, but the mean of all the predictions will be always in $(-5, 5)$.

For RCF, it is unnecessary to implement the Gibbs sampling since a simple gradient based unbounded optimization algorithm (e.g., the steepest gradient descent) is enough. To train the model, the penalty term is set to be 0.3 and the learning pace is set to be 0.01. Figure 4(c) shows the RMSE of every iteration of RCF for the training set and the residual plot is shown in Figure 4(d). Figure 5 shows the qq-plots of the residuals indicating that the residual of RCF is not normally distributed.



(a) The qq-plot of the residual against the standard normal for the training data



(b) The qq-plot of the residual against the standard normal for the test data

Figure 5: The residual plots of RCF for MovieLens data set.

Table 3: RMSE and MAE on test set of the MovieLens dataset.

	GGCF		RCF	
	RMSE	MAE	RMSE	MAE
2	0.9700	0.7606	1.0283	0.8318
3	0.9635	0.7560	1.0372	0.8388
5	0.9575	0.7509	1.0314	0.8336
10	0.9508	0.7435	1.0299	0.8320
30	0.9574	0.7453	1.0293	0.8320

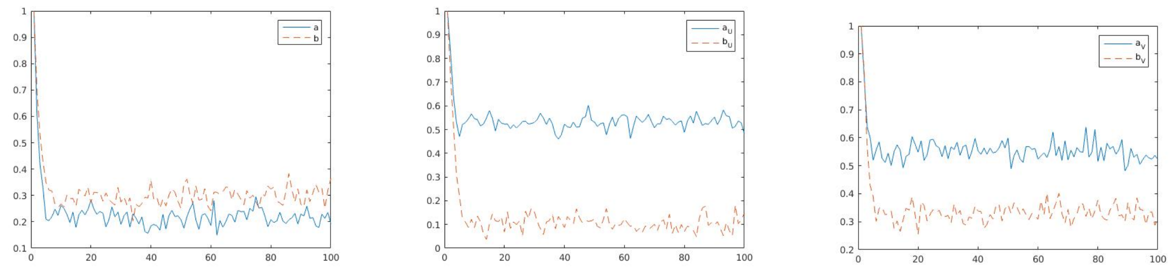
The results for test set of GGCF and RCF is shown in Table 3. By comparing the RMSE and MAE of the two models, we show that GGCF is slightly better than RCF in every dimension.

4.3. Book crossing dataset

The BookCrossing (BX) dataset was collected by Cai-Nicolas Ziegler in a 4-week crawl (August / September 2004) from the Book-Crossing community with kind permission from Ron Hornbaker, CTO of Humankind Systems. It contains 1,149,780 ratings given to 271,379 books from 278,858 users. The ratings have a scale from 1 to 10. We randomly select half of the ratings as training data and the rest are used as test data. Note we do not need to use 80% of the observations as training data since the size this dataset is much larger than the movieLens dataset. As a result, half of the observations contain most of the information. So we can test the model with more data just to give a more robust test error. The sparsity of this data set

Table 4: RMSE and MAE on test set of the Book crossing dataset.

	GGCF		RCF	
	RMSE	MAE	RMSE	MAE
2	0.8908	0.6226	1.109	0.8522
3	0.8989	0.6220	1.002	0.8212
5	0.8684	0.6109	0.9321	0.7801
10	0.8312	0.6135	0.9211	0.7821
30	0.8321	0.6053	0.9212	0.7324



(a) The parameters for $\lambda_{i,j}$. (b) The parameters for U_i. (c) The parameters for V_j.

Figure 6: The parameters versus iterations of GGCF for Book crossing dataset.

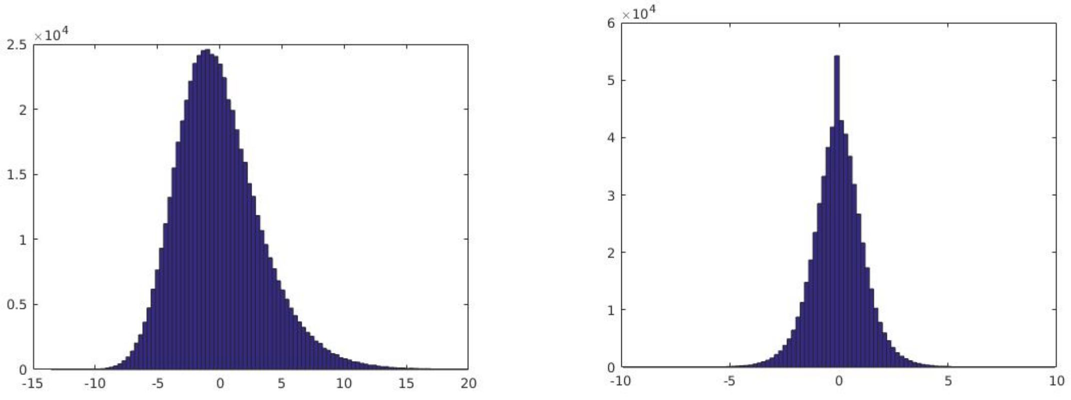
is obvious, because the observed ratings has a fraction less than 1% out of the possible ratings. Hence the Gaussian-Gamma distribution is a more reasonable assumption for both the ratings and the latent features.

In this experiment, we do not fix the parameters in the model but let them to be settled in the algorithm. The hierarchical Bayesian estimator is applied in the algorithm to set the parameters. We run the Gibbs sampling for 100 times for the parameters to settle down. In Figure 6 we show the values of the parameters in different iterations. It can be seen that the algorithm converges in about 10 iterations. In Figure 7 we show the residuals for the Book crossing dataset. In this dataset, GGCF also exhibits the heavy tail effect. Compared with the residual of RCF, the residual of GGCF clearly has more probabilities around 0. We also give the RMSE and the MAE for the Book crossing dataset of the two models and they are shown in Table 4. Again, GGCF outperforms RCF a bit.

5. Analytic analysis of the performance of GGCF

5.1. Convergence analysis of GGCF

In this subsection we give a theoretical analysis on the performance of GGCF which reflects the experimental results presented in Section 4. The purpose of the collaborative filtering model is to predict the



(a) The residual for RCF. (b) The residual for GGCF.

Figure 7: The residuals for Book crossing dataset.

unknown ratings with the inferred latent features. Hence, we need to show why the predicted value $\hat{R}_{i,j}$ converges to its true value. In Figure 4(a), we have seen the RMSE converges in some iterations. In this section, we want to give the reason by probability and statistical theory. Moreover, we have seen in Table 3 and Table 4 that GGCF has lower RMSE and MAE in both data sets, and we will give an explanation in this subsection.

Basically, we need to show the following statements. Firstly, the sample of the variables $\{\lambda_{i,j}, U_i, V_j, \lambda_{U_i}, \lambda_{V_j} : i = 1, ..., M, j = 1, ..., N\}$ provided by the inference part (Subsection 3.3) are drawn from their true distribution. Secondly, the predicted value $\hat{R}_{i,j}$ provided by equation (6) converges to the true value when T is large enough. Thirdly, we need to show that the proposed model GGCF always has better, or at least the same performance as the normal distribution based model, for example, the RCF.

The first statement holds by the convergence analysis of the Gibbs sampling method. To make things easier, we denote $\{\boldsymbol{\lambda}, \boldsymbol{U}, \boldsymbol{V}, \boldsymbol{\lambda}_U, \boldsymbol{\lambda}_V\} = \{\lambda_{i,j}, U_i, V_j, \lambda_{U_i}, \lambda_{V_j} : i = 1, ..., M, j = 1, ..., N\}$ and $\boldsymbol{R} = \{R_{i,j} : i = 1, ..., M, j = 1, ..., N\}$. We further use θ to represent the parameters in the model. Concretely, $\theta = \{a, b, a_U, a_V, b_U, b_V\}$. Since the true distribution

$$P(\boldsymbol{\lambda}, \boldsymbol{U}, \boldsymbol{V}, \boldsymbol{\lambda}_U, \boldsymbol{\lambda}_V | \boldsymbol{R}, \theta)$$

of the concerned variables is intractable. We can only sample these variables one by another using the conditional distributions provided by equation (4). That is, with some initial values of $\{\boldsymbol{\lambda}^{(1)}, \boldsymbol{U}^{(1)}, \boldsymbol{V}^{(1)}, \boldsymbol{\lambda}_U^{(1)}, \boldsymbol{\lambda}_V^{(1)}\}$, we sample $\lambda_{1,1}^{(2)}$ conditional on $\{\boldsymbol{\lambda}_{-(1,1)}^{(1)}, \boldsymbol{U}^{(1)}, \boldsymbol{V}^{(1)}, \boldsymbol{\lambda}_U^{(1)}, \boldsymbol{\lambda}_V^{(1)}\}$, where $\boldsymbol{\lambda}_{-(1,1)}^{(1)}$ denote the set $\boldsymbol{\lambda}^{(1)} \setminus \{\lambda_{1,1}\}$. Next, we sample $\lambda_{1,2}^{(2)}$ conditional on $\{\lambda_{1,1}^{(2)}, \boldsymbol{\lambda}_{-(1,1:2)}^{(1)}, \boldsymbol{U}^{(1)}, \boldsymbol{V}^{(1)}, \boldsymbol{\lambda}_U^{(1)}, \boldsymbol{\lambda}_V^{(1)}\}$, and then $\lambda_{1,3}^{(2)}$, and $\cdots$. The first iteration finishes when we reach the sample $\{\boldsymbol{\lambda}^{(2)}, \boldsymbol{U}^{(2)}, \boldsymbol{V}^{(2)}, \boldsymbol{\lambda}_U^{(2)}, \boldsymbol{\lambda}_V^{(2)}\}$. This procedure is just the Gibbs sampling inference, and its convergence analysis can be found in [30]. Hence when the number of iterations is set to be large enough, the empirical distribution formed by the sample approximates the true distribution.

In Figure 4(a) we show the Gibbs sampling converges after a few iterations. So we can use the converged sample to give an empirical distribution of the concerned variables.

The above analysis also justifies Figure 2. In the synthetic dataset, the observations are in fact the normal random variables with unknown standard deviations. When we add noise with different standard deviations, we expected the model has the ability to capture them in all cases. That is, we want the mean of the sampled $\lambda_{i,j}$ (the vertical coordinates) to be similar with the true setting (the horizontal coordinates). Shown in the previous paragraph, the sampled $\lambda_{i,j}$ by our algorithm converges to its true distribution. Hence its statistic (the mean) can be approximated by the mean of the sampled $\lambda_{i,j}$.

The second statement follows from the law of large numbers. In equation (6), $U_{i,t}$ and $V_{j,t}$ are sampled by the Gibbs sampling which produces a Markov chain. By the Monte Carlo Markov Chain (MCMC) version of the law of large numbers [31], $\hat{R}_{i,j}$ converges to its true value with probability 1 as T goes to infinity. That means when we have a sample of size large enough, the probability of $\hat{R}_{i,j}$ to not converge to its true value is 0. Hence the second statement follows.

Now we focus on the third statement that GGCF always beats RCF. This claim is clear when we notice the relationship between the $\mathcal{T}$ distribution and the normal distribution. It is well known that the $\mathcal{T}$ distribution converges to the normal distribution when the degree of freedom of the $\mathcal{T}$ distribution goes to infinite. In fact, when the degree of freedom is equal to or greater than 30, the density of the $\mathcal{T}$ and the normal distributions almost coincide. Hence we can resemble the normal distribution using a $\mathcal{T}$ distribution by adapting the degree of freedom. Since our approach, the Gaussian-Gamma distribution is in fact a two-parameters extension of the $\mathcal{T}$ distribution, it can also resemble the normal distribution by choosing the "right" parameters. In our model, the parameters of the Gaussian-Gamma distributions are a, b, a_U, b_U, a_V, b_V (see Figure 1). So when the data is normally distributed, GGCF can still capture its properties by adjusting the parameters to the desired values.

Moreover, when the data is not normally distributed, the Gaussian-Gamma distribution can still capture some of the properties while the normal distribution cannot. For example, Figure 5 shows the qq-plot of the MovieLens data set. Since the plot curves off in extreme values, we know that the underlying distribution is surely not a normal distribution. And it is most likely to be a heavy tail distribution. Hence we use a heavy tail distribution as the basic assumption of the observations. By adding a higher hierarchy in the precision, a Gaussian-Gamma distribution surely offers this opportunity to model the data which is more complex. That is the reason we use GGCF instead of RCF.

In the traditional RCF, the rating $R_{i,j}$ is assumed to follow the following distribution

$$R_{i,j} \sim \mathcal{N}(U_i' V_j, \lambda_{i,j}^{-1}).$$

And in GGCF, the distribution of $R_{i,j}$ is changed to

$$\begin{aligned} R_{i,j} &\sim \mathcal{N}(U_i'V_j, \lambda_{i,j}^{-1}), \\ \lambda_{i,j} &\sim \mathcal{G}(a, b). \end{aligned} \tag{8}$$

This distribution is demonstrated in the left part of Figure 1. Equivalently, by integrating out $\lambda_{i,j}$, the rating $R_{i,j}$ follows

$$R_{i,j} \sim \mathrm{GG}(a, b),$$

where GG represents the Gaussian-Gamma distribution. By the analysis mentioned in the previous paragraph, the model of GGCF surely has a better performance comparing to RCF when the parameters a and b are set to be the right values. Hence it is essential to give the right values of the parameters a, b. Fortunately, we do not need to set these parameters manually. Subsection 3.4 gives the details of how to adjust these parameters automatically. From Table 3 and Table 4, we can see that GGCF has lower average errors in both RMSE and MAE, which is compatible with our analysis.

5.2. A proof of the robustness of GGCF

In this subsection, we want to include a mathematical proof of the robustness of GGCF. Concretely, we want to show that GGCF is more robust than the Gaussian based collaborative filtering. In [32], the authors defined the word "robustness" in the following way. Let y_i be observations indexed by i ($1 \leq i \leq n$), and they have mean μ and covariance matrix Σ, where μ is a $v \times 1$ vector and Σ is a $v \times v$ positive definite matrix. The estimation of μ is robust in the sense that outlying cases with large Mahalanobis distance $\delta_i^2 = (y_i - \mu)'\Sigma^{-1}(y_i - \mu)$ are downweighted.

In the collaborative filtering, the rating $R_{i,j}$ is assumed to be the inner product of the latent features U_i and V_j. Now we fix the item features V_j and show GGCF is more robust in the estimation of U_i. We define the index set $C_i = \{j : R_{i,j} \text{ is observed}\}$ for later use.

Property 1. *The estimation of U_i is more robust in GGCF than the Gaussian based collaborative filtering when V_j ($j \in C_i$) is fixed.*

Proof. In the Gaussian based collaborative filtering, the rating $R_{i,j}$ is assumed to have the distribution $R_{i,j} \sim \mathcal{N}(U_i'V_j, \lambda^{-1})$. Hence the likelihood function of U_i is

$$\mathcal{L}_1(U_i) = \prod_{j \in C_i} \left(\frac{\lambda}{2\pi}\right)^{1/2} \exp\left[-\frac{\lambda}{2}(R_{i,j} - U_i'V_j)^2\right].$$

Taking gradient of the logarithm of $\mathcal{L}_1(U_i)$ gives

$$\frac{\partial \log \mathcal{L}_1(U_i)}{\partial U_i} = -\lambda \sum_{j \in C_i} (R_{i,j} - U_i'V_j)V_j.$$

Consequently, the estimation of U_i should satisfy the equation

$$\sum_{j \in C_i} (R_{i,j} - U_i' V_j) V_j = 0. \tag{9}$$

In GGCF, $R_{i,j}$ is assumed to have the distribution shown in equation (8), and thus the joint distribution of $R_{i,j}$ for $j \in C_i$ is

$$L(\{U_i, \lambda_{i,j} : j \in C_i\}) = \prod_{j \in C_i} \left(\frac{\lambda_{i,j}}{2\pi}\right)^{1/2} \exp\left[-\frac{\lambda_{i,j}}{2}(R_{i,j} - U_i' V_j)^2\right] \frac{b^a}{\Gamma(a)} \lambda_{i,j}^{a-1} \exp(-b\lambda_{i,j}).$$

Since we are only interested in the estimation of U_i, we marginalize out the latent variables $\lambda_{i,j}$ and produce

$$\mathcal{L}_2(U_i) = \prod_{j \in C_i} \left(\frac{1}{2\pi}\right)^{1/2} \frac{b^a}{\Gamma(a)} \frac{\Gamma(1/2 + a)}{[b + 1/2(R_{i,j} - U_i' V_j)^2]^{1/2+a}}.$$

Here we use the equality $\int b^a/\Gamma(a) x^{a-1} e^{-bx} dx = 1$. Taking gradient of the logarithm of $\mathcal{L}_2(U_i)$ gives

$$\frac{\partial \log \mathcal{L}_2(U_i)}{\partial U_i} = -\left(\frac{1}{2} + a\right) \sum_{j \in C_i} \frac{(R_{i,j} - U_i' V_j) V_j}{b + 1/2(R_{i,j} - U_i' V_j)^2}.$$

So the estimation of U_i should satisfy the equation

$$\sum_{j \in C_i} w_{i,j} (R_{i,j} - U_i' V_j) V_j = 0, \tag{10}$$

where $w_{i,j} = [b + 1/2(R_{i,j} - U_i' V_j)^2]^{-1}$ is the weight assigned to the rating $R_{i,j}$. Obviously, $w_{i,j}$ decreases with $\delta_{i,j}$ increases. Comparing equation (10) with (9) we can see that the outlying cases with large distance $\delta_{i,j}$ are downweighted. Furthermore, we can also see from equation (10) that a smaller parameter b gives a more robust model. This concludes the property. □

Similarly, we can fix the user features U_i and show GGCF is more robust in the estimation of V_j. From this property we can see that the unusual large (or small) rating $R_{i,j}$ has less influence on GGCF than on Gaussian based models.

6. Discussion

In the previous section, we give a detailed analysis of the performance of the Gaussian-Gamma distributions, which is imposed on $R_{i,j}$. That analysis is based on the theoretical probabilistic perspective of view. In this section, we want to discuss the effect of the variables λ_{U_i} and λ_{V_j} in the language of machine learning.

In machine learning community, λ_{U_i} and λ_{V_j} are called the penalty terms, or the regularity terms. They represent our belief or expectation about the model. From equation (5) we can see that when λ_{U_i} (or λ_{V_j}) approaches ∞, the matrix $\tilde{\Sigma}_{U_i}^{-1}$ (or $\tilde{\Sigma}_{V_j}^{-1}$) will be a constant multiplies an identity. It implies that we believe the components of U_i (or V_j) are independent and every component are highly concentrate on the mean. In some applications, this assumption can improve the generalization ability of the model, because it punishes the so called "over-fitting" effect. But by how much we should regularize the model changes case by case. So we need to adjust their values in the experiments.

However, in our model, λ_{U_i} and λ_{V_j} are updated by equation (4). Hence GGCF has the ability of avoiding the time consuming empirical setting procedure. Moreover, in RCF, since the penalty terms are set in advance, it is impossible to give every U_i a λ_{U_i} because there are so many of them. Thus the only possible setting is to give all of them the same value. But in GGCF, instead of the simplistic equal assumption, the precisions are different. That is because in the inference phrase, the precisions λ_{U_i} and λ_{V_j} are updated in each iteration with the last two conditional probabilities provided in (4) until convergence. This is another advantage of GGCF over RCF.

We also want to list a limitation of the GGCF in this section. It lies in the fact that both the normal distribution and the Gassian-Gamma distribution are symmetric. So when the distribution of the data is skewed, both GGCF and RCF are not appropriate. In that case, it should be better to impose a skewed distribution on the observations. For example, the Gamma distribution is a famous skewed continuous distribution and the Poisson distribution is a famous skewed discrete distribution. In real applications, if we observe skewness in the concerned data set, these distributions should be applied.

7. Conclusion

In this paper, we propose a Bayesian treatment of the traditional CF. By adding priors to the precisions of noise, the feature vectors, the model has shown to be more robust than a simple normal assumption. Moreover, by using these priors, the precisions no longer need to be set in a simplistic way by assuming that they are equal and known in advance. We also discuss how to set the parameters of the model. The method is to set them empirically by using small positive numbers to employ a large support or to set automatically using the hierarchical Bayes estimator. For inference, we give a multi-stage Gibbs sampler and demonstrate the statistical meaning of the conditionals of the variables. In our extensive experiments, we verify that the model is a more robust distribution in both the synthetic data set and the real datasets from the demonstration of the residual plots. The RMSE and MAE of GGCF and RCF for MovieLens and Book crossing datasets have been given to validate that our model is better than RCF. Our future work aims to develop an inference algorithm that can be implemented on a parallel platform.

8. Acknowledgment

This work was supported by the National Basic Research Program of China(2014CB340404), the National Natural Science Foundation of China(71571136), the Project of Science and Technology Commission of Shanghai Municipality (16JC1403000), and the IBM SUR (2015) grant.

References

[1] K. Y, B. R, V. C, Matrix Factorization Techniques for Recommender Systems, Computer 42 (8) (2009) 30–37.

[2] R. Salakhutdinov, A. Mnih, Probabilistic Matrix Factorization, in: Advances in Neural Information Processing Systems,, 2008, pp. 1257–1264.

[3] C. R. Rao, H. Toutenburg, C. Heumann, Principles and Theory for Data Mining and Machine Learning, Vol. 26, 2009. `arXiv:arXiv:1011.1669v3`, `doi:10.1007/978-0-387-98135-2`.
URL `http://www.springer.com/statistics/statistical+theory+and+methods/book/978-0-387-98134-5?cm{_}mmc=AD-{_}-Enews-{_}-ECS12245{_}V1-{_}-978-0-387-98134-5`

[4] C. D. Mining, H. Schütze, Foundations of statistical natural language processing, Cambridge: MIT press, 1999.

[5] N. Lawrence, Probabilistic non-linear Principal Component Analysis with Gaussian Process Latent Variable Models, Journal ofMachine Learning Research 6 (2005) 1783–1816.
URL `http://eprints.pascal-network.org/archive/00000914/`

[6] R. E. Schapire, Y. Freund, Boosting: Foundations and Algorithms, 2012. `arXiv:arXiv:1011.1669v3`.

[7] X. Zhou, K. Li, Y. Zhou, K. Li, Adaptive Processing for Distributed Skyline Queries over Uncertain Data ,, IEEE Transactions on Knowledge & Data Engineering 28 (2) (2016) 371–384.

[8] K. Li, C. Liu, K. Li, A. Y. Zomaya, A Framework of Price Bidding Configurations for Resource Usage in Cloud Computing, IEEE Transactions on Parallel and Distributed Systems 27 (8) (2016) 2168–2181.

[9] L. Zhang, K. Li, Y. Xu, J. Mei, F. Zhang, K. Li, Maximizing reliability with energy conservation for parallel task scheduling in a heterogeneous cluster, Information Sciences 319 (2015) 113–131.

[10] K. Li, X. Tang, B. Veeravalli, K. Li, Scheduling Precedence Constrained Stochastic Tasks on Heterogeneous Cluster Systems, IEEE Transactions on Computers 64 (1) (2015) 191–204.

[11] Y. Xu, K. Li, L. He, L. Zhang, K. Li, A Hybrid Chemical Reaction Optimization Scheme for Task Scheduling on Heterogeneous Computing Systems, IEEE Transactions Parallel Distributed Systems 26 (12) (2015) 3208–3222.

[12] K. Li, W. Yang, K. Li, Performance Analysis and Optimization for SpMV on GPU Using Probabilistic Modeling, IEEE Transactions on Parallel and Distributed Systems 26 (1) (2015) 196–205.

[13] C. P. Robert, G. Casella, Monte Carlo Statistical Methods., Springer, 2009.

[14] R. Kannan, M. Ishteva, H. Park, Bounded matrix factorization for recommender system, Knowledge and Information Systems 39 (3) (2014) 491–511.

[15] N. Srebro, T. Jaakkola, Weighted low-rank approximations, in: ICML, 2003, pp. 720–727.

[16] H. Ma, An experimental study on implicit social recommendation, in: International ACM SIGIR Conference on Research and Development in Information Retrieval, 2013.

[17] M. Jamali, M. Ester, A matrix factorization technique with trust propagation for recommendation in social networks, in: Proceedings of the fourth ACM conference on Recommender systems, 2010, pp. 135–142.

[18] K. P. Murphy, Machine learning: a probabilistic perspective, MIT press, 2012.

[19] C. M. Bishop, Pattern recognition and machine learning, springer, 2006.

[20] J. E. Griffin, P. J. Brown, Inference with normal-gamma prior distributions in regression problems, ,, Bayesian Analysis 5 (1) (2010) 171–188.

[21] M. E. Tipping, N. D. Lawrence, Variational inference for Student-t models: Robust Bayesian interpolation and generalised component analysis, Neurocomputing 69 (1-3) (2005) 123–141.

[22] M. M. E. Boggis, K. Walters, Exploiting adaptive bayesian regression shrinkage to identify exome sequence variants associated with gene expression, in: The Contribution of Young Researchers to Bayesian Statistics, Springer, 2014, pp. 135–138.

[23] E. C. Neto, I. Jang, S. H. Friend, A. A. Margolin, The stream algorithm: computationally efficient ridge-regression via bayesian model averaging, and applications to pharmacogenomic prediction of cancer cell line sensitivity,, in: Pac. Symp. Biocomput, 2014, pp. 27–38.

[24] H. Bae, T. Perls, M. Steinberg, P. Sebastiani, Bayesian polynomial regression models to fit multiple genetic models for quantitative traits, Bayesian analysis (Online) 10 (1) (2015) 53.

[25] S. Funk, Netflix Update: Try This at Home (2006).
URL `http://sifter.org/{~}simon/journal/20061211.html`

[26] A. Paterek, Improving regularized singular value decomposition for collaborative filtering, in: Proceedings of KDD cup and workshop, 2007, pp. 5–8.

[27] Y. Zhou, D. Wilkinson, R. Schreiber, R. Pan, Large-scale parallel collaborative filtering for the netflix prize, in: Algorithmic Aspects in Information and Management, Springer, 2008, pp. 337–348.

[28] G. Casella, E. I. George, Explaining the gibbs sampler, The American Statistician 46 (3) (1992) 167–174.

[29] A. Bjorck, Numerical methods for least squares problems, Society for Industrial & Applied Mathematics 21 (2) (1996) 1081–1098.

[30] G. O. Roberts, A. F. M. Smith, Simple conditions for the convergence of the Gibbs sampler and Metropolis-Hastings algorithms, Stochastic Processes and their Applications 49 (2) (1994) 207–216. `doi:10.1016/0304-4149(94)90134-1`.

[31] G. O. Roberts, J. S. Rosenthal, General state space Markov chains and MCMC algorithms., Probability Surveys 1 (2004) 20–71. `arXiv:0404033v4`, `doi:10.1214/154957804100000024`.
URL `http://arxiv.org/abs/math/0404033`

[32] K. L. Lange, R. J. A. Little, J. M. G. Taylor, Robust statistical modeling using the t distribution, Journal of the American Statistical Association 84 (408) (1989) 881–896. `doi:10.1080/01621459.1989.10478852`.