Sparse Bayesian Content-Aware Collaborative Filtering for Implicit Feedback

Defu Lian[1], **Yong Ge**[2], **Nicholas Jing Yuan**[3], **Xing Xie**[4], **Hui Xiong**[5]
[1]Big Data Research Center, University of Electronic Science and Technology of China
[2]University of North Carolina at Charlotte, [3]Microsoft Corporation
[4]Microsoft Research, [5]Rutgers University
{dove.ustc,ygestrive} at gmail.com, {nichy,xingx} at microsoft.com, hxiong at rutgers.edu

Abstract

The popularity of social media creates a large amount of user-generated content, playing an important role in addressing cold-start problems in recommendation. Although much effort has been devoted to incorporating this information into recommendation, past work mainly targets explicit feedback. There is still no general framework tailored to implicit feedback, such as views, listens, or visits. To this end, we propose a sparse Bayesian content-aware collaborative filtering framework especially for implicit feedback, and develop a scalable optimization algorithm to jointly learn latent factors and hyperparameters. Due to the adaptive update of hyperparameters, automatic feature selection is naturally embedded in this framework. Convincing experimental results on three different implicit feedback datasets indicate the superiority of the proposed algorithm to state-of-the-art content-aware recommendation methods.

1 Introduction

Recommender systems provide users with personalized recommendations for items (e.g., products and services), hopefully suiting their unique needs. In the past years, they have been used for functions like increasing sales in e-commerce and clicking rates on websites, and improving general visitor satisfaction, and thus are attracting a lot of attention from academia and industry [Linden *et al.*, 2003; Adomavicius and Tuzhilin, 2005; Koren, 2008]. Typically, these systems profile users and items from the implicitly or explicitly generated data and recommend user the most related items. Due to the large amount of implicit feedback, such as views, listening, or visiting behaviors, and the specific characteristics of lacking substantial evidence on which items that users dislike (i.e., negative items), recommendation for implicit feedback datasets has already become an important research direction [Hu *et al.*, 2008; Pan *et al.*, 2008; Rendle *et al.*, 2009; Gopalan *et al.*, 2015].

The popularity of social media makes the availability of a large amount of user-generated content and user information, playing an important part in addressing the cold-start problems in recommendation. Although much effort has been devoted to incorporating content into recommendation, such as LibFM [Rendle, 2012], MatchBox [Stern *et al.*, 2009], regression-based latent factor model [Agarwal and Chen, 2009] and MF-EFS [Koenigstein and Paquet, 2013], they are mainly designed for explicit feedback. Although recommendation for implicit feedback has been integrated with topic modeling for incorporating items' text-content [Wang and Blei, 2011; Gopalan *et al.*, 2014], there is still no general framework tailored to implicit feedback to take any type of features. Applying the previously developed general algorithms for explicit feedback requires randomly sampling negative items from missing entries in the user-item matrix for better learning efficiency. However, such a strategy is empirically sub-optimal, compared to treating all missing entries as negative but assigning them a lower confidence, according to [Wang and Blei, 2011; Lian *et al.*, 2014; Liu *et al.*, 2014]. The varying confidence for negative and positive preference can also be modeled by a two stage process recommendation model, according to [Gopalan *et al.*, 2015], the first of which appropriately fit the volume of user activity, i.e., budget. However, their evaluation doesn't indicate a strong advantage on implicit feedback. Another alternative modeling approach for varying confidence is to leverage ranking-based factorization algorithms, especially optimizing metrics at the top positions [Weston *et al.*, 2010]. Nevertheless, it does not perform as well as expected empirically according to our evaluation results, and even suffered from computation issues [Li *et al.*, 2015].

Due to the effectiveness and efficiency of [Hu *et al.*, 2008], we propose a content-aware collaborative filtering framework tailored to implicit feedback based on their algorithm. It can incorporate any kind of content of users and items at the same time of both steering clear of negative sampling and modeling the varying confidence. It takes a user-item preference matrix, a user-feature matrix (e.g., gender, age) and an item-feature matrix (e.g., categories, descriptions) as input and maps each user, each item, and each feature of both users and items into a joint latent space. This framework can exploit gradient descent for parameter learning, but suffers from time-consuming search (e.g., grid search) for the learning rate and the regularization coefficients.

To address this issue, we first present an equivalent but more generalized probabilistic generative model for implicit feedback, where both latent factors and hyperparameters (i.e.,

regularization coefficients) are considered hidden variables. Based on this model, we develop a scalable optimization algorithm for jointly learning latent factors and hyperparameters. Due to the adaptive update of hyperparameters, automatic relevance determination and even feature selection for content is naturally embedded in this framework according to the sparse Bayesian learning framework [Tipping, 2001]. This also leads to another distinguishing characteristic compared to Libfm, svdfeature [Chen *et al.*, 2012] and et al., in addition to steering clear of negative sampling based on a highly efficient learning algorithm.

The proposed algorithm is then evaluated on three different implicit feedback datasets and compared with several state-of-the-art content-aware recommendation algorithms, including Libfm and svdfeature. Convincing experimental results indicate that the proposed algorithm is not only capable of pruning non-informative features, but also best tailored to content-aware recommendation for implicit feedback.

2 Preliminary

2.1 Matrix Factorization for Implicit Feedback

Matrix factorization for implicit feedback operates a user-item preference matrix $\mathbf{R} \in \{0,1\}^{M \times N}$, including M users and N items. Each entry $r_{i,j} \in \mathbf{R}$ indicates whether the entry is observed or not, where i and j are reserved for indexing users and items, respectively. In matrix factorization, users and items are represented in a joint latent space of dimension K: user i is represented by a latent vector $\mathbf{s}_i \in \mathbb{R}^K$ and item j by a latent vector $\mathbf{t}_j \in \mathbb{R}^K$. We form the predicting preference of user i for item j with an inner product, i.e., $\hat{r}_{i,j} = \mathbf{s}_i' \mathbf{t}_j$.

Due to the lack of negative preference in implicit feedback, matrix factorization in this case needs to either sample pseudo-negative items for each user from missing entries in the user-item matrix [Pan *et al.*, 2008; Rendle *et al.*, 2009] or treat the data as an indication of positive and negative preference with vastly varying confidence [Hu *et al.*, 2008]. Due to empirically observed superiority, we focus on the latter algorithm [Lian *et al.*, 2014; Liu *et al.*, 2014], which minimizes the weighted squared loss with a regularized term:

$$\mathcal{L} = \frac{1}{2} \sum_{i,j} w_{i,j} (r_{i,j} - \mathbf{s}_i' \mathbf{t}_j)^2 + \frac{\lambda}{2} (\sum_i \|\mathbf{s}_i\|^2 + \sum_j \|\mathbf{t}_j\|^2),$$

where λ is a regularization coefficient and $w_{i,j}$ indicates the confidence of user preference, i.e.,

$$w_{i,j} = \begin{cases} \beta & \text{if } (i,j) \text{ observed} \\ 1 & \text{otherwise} , \end{cases}$$

where β is a tuning parameter satisfying $\beta \gg 1$, so that the confidence of the observed entries are much larger than the missing ones.

However, such a matrix factorization will fail in the cold-start problem. A general solution is to integrate collaborative filtering with content-based filtering [Pazzani, 1999] and has been yielded by some popular content-aware collaborative filtering frameworks, such as LibFM, MatchBox, and SVDFeature. However, they mainly target explicit feedback datasets, which include both positively and negatively preferred samples. Feeding implicit feedback into them is presumed to be suboptimal, due to the empirically observed superiority of the above algorithm for implicit feedback. This thus motivates us to develop a general content-aware collaborative filtering framework tailored to implicit feedback.

2.2 Content-Aware Collaborative Filtering for Implicit Feedback

In content-aware collaborative filtering frameworks, in addition to users and items, their features are also represented as latent vectors in the joint latent space [Stern *et al.*, 2009; Chen *et al.*, 2012; Rendle, 2012]. Assuming users have F features and items have L features, the predicting preference of a user i for an item j is then formulated as $\hat{r}_{i,j} = (\mathbf{s}_i + \mathbf{U}'\mathbf{x}_i)'(\mathbf{t}_j + \mathbf{V}'\mathbf{y}_j)$, where $\mathbf{U} \in \mathbb{R}^{F \times K}$ is a latent matrix of user features and $\mathbf{V} \in \mathbb{R}^{L \times K}$ is a latent matrix of item features.

If the ids of both users and items are also considered features and encapsulated into $\{\tilde{\mathbf{x}}_i\}$ and $\{\tilde{\mathbf{y}}_j\}$, the predicting preference is simplified as $\hat{r}_{i,j} = \tilde{\mathbf{x}}_i' \tilde{\mathbf{U}} \tilde{\mathbf{V}}' \tilde{\mathbf{y}}_j$, where $\tilde{\mathbf{U}} \in \mathbb{R}^{(M+F) \times K}$ is obtained by concatenating $\{\mathbf{s}_i\}$ and $\mathbf{U}$ by rows ($\tilde{\mathbf{V}}$ shares a similar meaning). Then, as proposed in [Lian *et al.*, 2015], the objective function is formulated as:

$$\mathcal{L} = \frac{1}{2} \sum_{i,j} w_{i,j} (r_{i,j} - \tilde{\mathbf{x}}_i' \tilde{\mathbf{U}} \tilde{\mathbf{V}}' \tilde{\mathbf{y}}_j)^2 + \frac{\lambda}{2} (\|\tilde{\mathbf{U}}\|_F^2 + \|\tilde{\mathbf{V}}\|_F^2).$$

Although we can use gradient descent directly for optimization w.r.t $\tilde{\mathbf{U}}$ and $\tilde{\mathbf{V}}$, it is more appealing to apply alternative least square due to the removal of the tuned learning rate. However, by simple algebra, we find the analytic solutions of latent vectors of different features are coupled and thus unable to compute efficiently. By rewriting the objective function, according to [Lian *et al.*, 2015], the overall optimization can be split into two stages, i.e.,

$$\begin{aligned} \mathcal{L} = & \frac{1}{2} \sum_{i,j} w_{i,j} (r_{i,j} - \mathbf{p}_i' \mathbf{q}_j)^2 + \frac{\lambda}{2} \sum_i \|\mathbf{p}_i - \mathbf{U}'\mathbf{x}_i\|^2 \\ & + \frac{\lambda}{2} \sum_i \|\mathbf{q}_i - \mathbf{V}'\mathbf{y}_i\|^2 + \frac{\lambda}{2} (\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2). \quad (1) \end{aligned}$$

The first stage learns summed latent vectors (i.e., $\mathbf{p}_i \triangleq \tilde{\mathbf{U}}'\tilde{\mathbf{x}}_i$ and $\mathbf{q}_j \triangleq \tilde{\mathbf{V}}'\tilde{\mathbf{y}}_j$); the second stage learns the feature latent matrices, $\mathbf{U}$ and $\mathbf{V}$. Both stages have analytical solutions.

3 Sparse Bayesian Content-Aware Collaborative Filtering for Implicit Feedback

Although user/item content can be incorporated by the above framework, it suffers from the following three problems. First, when deriving the analytical solution for the feature latent matrix, we discover the disadvantage of only using a single regularization coefficient λ. Second, the regularization parameters should be tuned carefully, otherwise this approach is prone to overfitting. Third, the relevance of different

features should be differentiated from each other, and even some of them are fully irrelevant and should be discarded. To address these shortcomings, we introduce a sparse Bayesian learning algorithm, which places priors on the hyperparameters so that they can be learned directly without manually tuning. Due to the adaptive update of hyperparameters, automatic feature selection is naturally embedded within this framework.

3.1 Generative Model

Before introducing a sparse Baysian learning algorithm, we first generalize the above content-aware collaborative filtering framework as a probabilistic generative model. In particular, the preference $r_{i,j}$ of the matrix $\mathbf{R}$ is independently generated by Gaussian distributions,

$$Pr(\mathbf{R}|\mathbf{P},\mathbf{Q}) = \prod_{i,j} N(r_{i,j}|\mathbf{p}_i'\mathbf{q}_j, w_{i,j}^{-1}). \quad (2)$$

The prior distributions over the user/item latent factors are assumed to be Gaussian:

$$Pr(\mathbf{P}|\mathbf{U},\lambda_P) = \prod_{i=1}^{M} N(\mathbf{p}_i|\mathbf{U}'\mathbf{x}_i, \lambda_P^{-1}), \quad (3)$$

$$Pr(\mathbf{Q}|\mathbf{V},\lambda_Q) = \prod_{j=1}^{M} N(\mathbf{q}_j|\mathbf{V}'\mathbf{y}_j, \lambda_Q^{-1}), \quad (4)$$

where λ_P and λ_Q are both precisions of Gaussian distributions, determining the importance of user/item features in learning interaction between users and items. On these two precisions, we place a Gamma distribution prior with shape a and rate b, to capture prior knowledge of feature importance. For convenience, we assume that λ_P and λ_Q are distributed over a logarithmic scale, as suggested in [Tipping, 2001].

The prior distributions over the latent matrix of user/item features are assumed to be matrix normal, conjugate priors:

$$Pr(\mathbf{U}|\boldsymbol{\alpha}_U) = \mathcal{MN}_{F,K}(\mathbf{U}|0, \mathrm{diag}(\boldsymbol{\alpha}_U)^{-1}, \mathbf{I}_K), \quad (5)$$

$$Pr(\mathbf{V}|\boldsymbol{\alpha}_V) = \mathcal{MN}_{L,K}(\mathbf{V}|0, \mathrm{diag}(\boldsymbol{\alpha}_V)^{-1}, \mathbf{I}_K), \quad (6)$$

$\boldsymbol{\alpha}_U \in \mathbb{R}_+^F, \boldsymbol{\alpha}_V \in \mathbb{R}_+^L$ are vectors of non-negative hyperparameters governing the prior precision for each dimension of the feature latent matrix. Matrix normal distribution is a generalization of the multivariate normal distribution to matrix-valued random variables and described by a row based variance Σ_R and a column based variance Σ_C in addition to mean. The second order central moments of matrix normal distribution can be represented as $E[(\mathbf{z}-\boldsymbol{\mu})(\mathbf{z}-\boldsymbol{\mu})'] = \boldsymbol{\Sigma}_R tr(\boldsymbol{\Sigma}_C)$ and $E[(\mathbf{z}-\boldsymbol{\mu})'(\mathbf{z}-\boldsymbol{\mu})] = \boldsymbol{\Sigma}_C tr(\boldsymbol{\Sigma}_R)$. Here, we don't explicitly place independent Gamma prior on $\boldsymbol{\alpha}_U$ and $\boldsymbol{\alpha}_V$, since the improper prior (shape and rate parameter approaching to zero) could incur the maximal sparsity.

Due to the conjugation, we can directly marginalize them out, leading to matrix normal distributed user/item latent factors with zero mean and a content-dependent covariance. For example, the marginal distribution over user latent factors $\mathbf{P}$ has the following form:

$$Pr(\mathbf{P}|\boldsymbol{\alpha_U},\lambda_P) = \mathcal{MN}_{M,K}(\mathbf{P}|0, \boldsymbol{\Sigma}_P, \mathbf{I}_K) \quad (7)$$

where $\boldsymbol{\Sigma}_P = \lambda_P^{-1}\mathbf{I}_M + \mathbf{X}\mathbf{A}_U^{-1}\mathbf{X}'$ and $\mathbf{A}_U = \mathrm{diag}(\boldsymbol{\alpha_U})$. In other words, the content information actually constrains the similarity of latent factors between users and between items. Therefore, if the content of two users/items is similar, in terms of a metric related to their normalized dot product, their latent factors should also be proximal in the latent space. However, different from previous work [Zhou *et al.*, 2012], which assumed content-based similarity between users or between items is given in advance and thus independent to this generative model, this similarity metric is adaptive with the update of the parameter $\boldsymbol{\alpha_U}$ and naturally embeds the capability of feature selection. For example, when the prior precision α_U^f for the f^{th} feature of user content increasingly approaches to infinity, its contribution in $\boldsymbol{\Sigma}_P$ (i.e., measuring the content-based similarity) becomes smaller until it is fully neglected. Below, we will study how to learn hyperparameters under the framework of evidence maximization or type II maximum likelihood in addition to deriving the update formula of latent factors of users/items and their features. For convenience, we denote $\{\mathbf{P},\mathbf{Q},\boldsymbol{\alpha}_U,\boldsymbol{\alpha}_V,\lambda_P,\lambda_Q\}$ as Θ to learn.

3.2 Parameter Learning

Based on the generative model, combining Eq (7) and Eq (2) with prior distributions of λ_P and λ_Q, we can derive the following objective function:

$$\begin{aligned}\mathcal{L}(\Theta) = &\sum_{i,j} w_{i,j}(r_{i,j} - \mathbf{p}_i'\mathbf{q}_j)^2 + K\log|\boldsymbol{\Sigma}_P| \\ &+ tr(\mathbf{P}'\boldsymbol{\Sigma}_P^{-1}\mathbf{P}) + K\log|\boldsymbol{\Sigma}_Q| + tr(\mathbf{Q}'\boldsymbol{\Sigma}_Q^{-1}\mathbf{Q}) \\ &- 2a(\log\lambda_P + \log\lambda_Q) + 2b(\lambda_P + \lambda_Q) \quad (8)\end{aligned}$$

Since each parameter for minimizing $\mathcal{L}(\Theta)$ cannot be obtained in a closed form given other fixed, it is possible to directly exploit gradient descent based alternative least square for parameter learning, but it suffers from the difficulty of selecting an appropriate learning rate.

For the sake of learning parameters more efficiently and encouraging a sparse solution of the feature latent factors, we resort to Bayesian Occam's razor [MacKay, 1992], to obtain iterative re-estimation for the parameters and the hyperparameters. Bayesian Occam's razor approximates the evidence, e.g., $Pr(\mathbf{P}|\boldsymbol{\alpha}_U,\lambda_P)$ by $Pr(\mathbf{P}|\mathbf{U}_{\text{MP}},\lambda_P)Pr(\mathbf{U}_{\text{MP}}|\boldsymbol{\alpha}_U)\Delta\mathbf{U}$. Since $\mathbf{U}$ follows the matrix normal distribution, such an approximation has an extract form. In particular, the extract form can be obtained by noting that

$$\begin{aligned}tr(\mathbf{P}'\boldsymbol{\Sigma}_P^{-1}\mathbf{P}) = &tr(\lambda_P(\mathbf{P}-\mathbf{X}\mathbf{U}_{\text{MP}})'(\mathbf{P}-\mathbf{X}\mathbf{U}_{\text{MP}})) \\ &+ tr(\mathbf{U}_{\text{MP}}'\mathbf{A}_U\mathbf{U}_{\text{MP}}), \quad (9)\end{aligned}$$

and expressing $|\boldsymbol{\Sigma}_P|$ as, using the matrix inversion lemma,

$$|\boldsymbol{\Sigma}_P| = |\boldsymbol{\Sigma}_U^{-1}||\lambda_P\mathbf{I}_M|^{-1}|\mathbf{A}_U|^{-1}, \quad (10)$$

where $\boldsymbol{\Sigma}_U^{-1} = (\lambda_P\mathbf{X}'\mathbf{X} + \mathbf{A}_U)$ and $\mathbf{U}_{\text{MP}} = \lambda_P\boldsymbol{\Sigma}_U\mathbf{X}'\mathbf{P}$. Substituting Eq (9) and Eq (10) into Eq (8), and applying alternative least square for optimization, we can get update equations of each parameter in the set Θ.

Below, we only study how to update user-related parameters because of the symmetry of the updating rules between

user-related parameters $(\mathbf{P}, \boldsymbol{\alpha}_U, \lambda_P)$ and item-related parameters $(\mathbf{Q}, \boldsymbol{\alpha}_V, \lambda_Q)$. In particular, computing the gradient of $\mathcal{L}(\Theta)$ w.r.t $\mathbf{p}_i$ and setting them to zero, we can obtain,

$$\mathbf{p}_i = (\mathbf{Q}'\mathbf{W}_i\mathbf{Q} + \lambda_P\mathbf{I}_K)^{-1}(\mathbf{Q}'\mathbf{W}_i\mathbf{r}_i + \lambda_P\mathbf{U}'_{\text{MP}}\mathbf{x}_i), \quad (11)$$

where $\mathbf{W}_i = \text{diag}([w_{i,1}, \cdots, w_{i,N}])$ and $\mathbf{r}_i$ is the i^{th} row of the preference matrix $\mathbf{R}$. Thus, given $\mathbf{Q}$ fixed, the update of latent factor for each user is independent to other users, making it straightforward to update the latent factors for all users in parallel. Similarly, deriving the gradient of $\mathcal{L}(\Theta)$ with respect to λ_P and α_U^f, and setting them to zero, we can obtain the updating formulas for them,

$$\lambda_P = \frac{K(M - \sum_f \gamma_U^f) + 2a}{tr((\mathbf{P} - \mathbf{X}\mathbf{U}_{\text{MP}})'(\mathbf{P} - \mathbf{X}\mathbf{U}_{\text{MP}})) + 2b} \quad (12)$$

$$\alpha_U^f = \frac{K\gamma_U^f}{|\mathbf{u}_{\text{MP}}^f|_2^2} \quad (13)$$

where $\mathbf{u}_{\text{MP}}^f$ represents the f^{th} row of user feature latent matrix $\mathbf{U}_{\text{MP}}$ and $\gamma_U^f = 1 - \alpha_U^f \Sigma_U^{f,f}$, which is suggested in MacKay fixed-point update rules [MacKay, 1992]. The latter quantity γ_U^f ranges from zero to one, measuring how well-determined its corresponding parameter $\mathbf{u}_f$ by the user content. The γ_U^f of all user features are concatenated as a vector, then $\boldsymbol{\gamma}_U = \text{diag}(\mathbf{I} - \boldsymbol{\Sigma}_U\mathbf{A}_U) = \lambda_P\text{diag}(\boldsymbol{\Sigma}_U\mathbf{X}'\mathbf{X})$. The overall algorithm is given in Algorithm 1.

Due to $tr(\mathbf{P}'\boldsymbol{\Sigma}_P^{-1}\mathbf{P}) \leq tr(\lambda_P(\mathbf{P} - \mathbf{X}\mathbf{U})'(\mathbf{P} - \mathbf{X}\mathbf{U})) + tr(\mathbf{U}'\mathbf{A}_U\mathbf{U})$ according to Eq (9), the derived updating rules actually optimize an upper bound of Eq (8) with an additional parameter $\mathbf{U}$, which replaces the left-hand side of the inequality with the right-hand side. Thus, the convergence of the updating rules can be guaranteed according to the convergence study of alternative least square [Grippo and Sciandrone, 2000] and the convergence study of sparse Bayesian learning [Wipf and Rao, 2004].

3.3 Complexity Analysis

Applying the trick used in [Hu *et al.*, 2008], the overall cost of computing user/item latent factors is $\mathcal{O}(\|\mathbf{R}\|_0K^2 + (M + N)K^3)$. Due to the independence of computation among different users/items, the actual cost can be reduced based on simple task parallel techniques. The total time complexity for computing $\mathbf{U}_{\text{MP}}$ and $\mathbf{V}_{\text{MP}}$ is $\mathcal{O}((\|\mathbf{X}\|_0 + \|\mathbf{Y}\|_0)K\#iter)$ since we can apply any iterative algorithm, such as conjugate gradient descent, for linear equations system, where $\#iter$ is the maximum number of iterations for convergence. The major cost in this algorithm is to compute the $\boldsymbol{\gamma}_U$ and $\boldsymbol{\gamma}_V$, which requires $\mathcal{O}((\|\mathbf{X}\|_0F + \|\mathbf{Y}\|_0L)\#iter)$ in the worst case in total since it also involves solving linear equations system. In practical, the time complexity also depends on the sparse pattern of the feature matrices in addition to the number of non-zero entries, thus it is hard to determine time complexity exactly, but it is obvious that the algorithm will be much more efficient when the feature matrices have special sparse structures. For the sake of more efficient parameter learning, we suggest performing dimension reduction, such as SVD and LDA, before feeding the feature matrices.

Algorithm 1: Sparse Bayesian Collaborative Filtering

Input : rating matrix $\mathbf{R}$, dimension K of latent space, user feature matrix $\mathbf{X}$, item feature matrix $\mathbf{Y}$, confidence β for observed events, shape a and rate b of gamma prior

Output: user latent factor $\mathbf{P}$ and item latent factor $\mathbf{Q}$

```
Initialize U_MP, V_MP with zero, and randomly initialize P and Q;
repeat
    Update each user's latent factor based on Eq (11);
    Update each item's latent factor similar to Eq (11);
    repeat
        Compute U_MP and γ_U;
        Update λ_P based on Eq (12);
        Update α_U based on Eq (13);
    until α_U and λ_P is convergent;
    repeat
        Compute V_MP and γ_V;
        Update λ_Q similar to Eq (12);
        Update α_V similar to Eq (13);
    until α_V and λ_Q is convergent;
until L(Θ) is convergent;
```

3.4 Induced Sparsity

In the case of the above setting, feature selection is naturally embedded in this framework. Let's inspect this property below, studying the marginal prior distribution over user feature factor $\mathbf{U}$ after marginalizing over the precision $\boldsymbol{\alpha}_U$ ($\boldsymbol{\alpha}_U \sim \prod_f Ga(\alpha_U^f|c, d)$). After an integral computation, we observe it follows an independent multivariate Student's t distribution, i.e. $Pr(\mathbf{U}|c, d) = \prod_f \mathcal{T}_{2c}(\mathbf{u}_f|0, \frac{d}{c}\mathbf{I}_K)$. This distribution is proportional to $\prod_f |\mathbf{u}_f|_2^{-K}$ when c and d approach zero, encouraging a row-wise sparsity on $\mathbf{U}$.

However, instead of marginalizing over $\mathbf{U}$, the marginalization over the hyperparameters is not consistent with the proposed optimization algorithm. To this end, based on [Wipf and Nagarajan, 2008], we seek a tight upper bound of $\log|\boldsymbol{\Sigma}_P|$ based on the concavity with respect to $\boldsymbol{\alpha}_U^{-1} = [1/\alpha_U^1, \cdots, 1/\alpha_U^F]$. In particular, we can get

$$\log|\boldsymbol{\Sigma}_P| = \min_{\mathbf{z}} \mathbf{z}'\boldsymbol{\alpha}_U^{-1} - g^*(\mathbf{z}) \quad (14)$$

where $g^*(\mathbf{z})$ is the concave conjugate of $\log|\boldsymbol{\Sigma}_P|$ w.r.t $\boldsymbol{\alpha}_U^{-1}$. Combining it with the upper bound of $tr(\mathbf{P}'\boldsymbol{\Sigma}_P^{-1}\mathbf{P})$, and optimizing over $\boldsymbol{\alpha}_U$ first, we can get $\alpha_U^f = \sqrt{z_f}/|\mathbf{u}_f|$ exactly. At this moment, $tr(\mathbf{U}'\mathbf{A}_U\mathbf{U})$, a part of the upper bound of $tr(\mathbf{P}'\Sigma_P^{-1}\mathbf{P})$, can be rewritten as $\sum_f \sqrt{z_f}|\mathbf{u}_f|_2$. When all z_f equal to one, it is just an $\ell_{2,1}$ norm for $\mathbf{U}$. Since this procedure also optimizes over a variational parameter z_U^f, it actually corresponds to an adaptive $\ell_{2,1}$ regularization.

4 Experiments

We evaluated our model on three tasks, which recommends locations on location-based social networks, articles on researcher communities, and music on online music searching

Table 1: Dataset statistics. Each user in Jiepang has age and gender info, and each item has two levels of category hierarchy; each user in Last.fm has age, gender, and country info; each CiteULike article is represented by learned 200 topics.

dataset	#records	#users	#items	density
Jiepang	1,521,105	28,050	27,405	0.00197
CiteULike	204,986	5,551	16,980	0.00217
Last.fm	819,995	981	107,397	0.00778

and browsing services. For the first task, we used self-crawled human mobility data from Jiepang, a Chinese location-based social network, and collected users' profiles from Sina Weibo (China's Twitter), since Jiepang users can log in with Weibo accounts. For the latter two tasks, we used public datasets, that is, the CiteULike dataset used in [Wang and Blei, 2011] and the Last.fm provided by [Celma, 2010], respectively. In these three datasets except CiteULike, we filtered items on which fewer than 10 users having actions (check-in, listening) and filtered users who have actions on fewer than 10 items. The CitULike dataset was kept unchanged to directly compare our model with the algorithm of [Wang and Blei, 2011]. The statistics after filtering are shown in Table 1. Based on three datasets, we performed 5-fold cross validation. For every user, we evenly split their user-item pairs into 5 folds. For each fold, we trained a model on other folds of data and tested it on the within-fold items for each user.

4.1 Metric

We evaluated the recommendation algorithms on sets of held-out observed entries. Presenting each user with the top k candidate items sorted by their predicting preference, we assessed recommendation performance by two widely used metrics, $recall@k$ and $precision@k$ [Wang and Blei, 2011]:

$$recall@k = \frac{1}{M}\sum_{u=1}^{M}\frac{|\mathbb{S}_u(k)\cap\mathbb{V}_u|}{|\mathbb{V}_u|}$$

$$precision@k = \frac{1}{M}\sum_{u=1}^{M}\frac{|\mathbb{S}_u(k)\cap\mathbb{V}_u|}{k},$$

where $\mathbb{S}_u(k)$ is the collection of top k recommended items for a user u and $\mathbb{V}_u$ is the set of the observed entries for her.

4.2 Competing Algorithms

We compared our model, named SBICCF, with the following five competing baselines on these three recommendation tasks. SBICCF was configured by manually tuning the confidence β of positive feedback. For location and article recommendation, $\beta = 30$; for music recommendation, $\beta = 1$. We can see that the latter β is much smaller than others, probably due to much higher density of the dataset.

1) **Libfm** [Rendle, 2012], supports classification and regression tasks, but requires drawing negative examples when being applied for recommendation on implicit feedback. We chose a MCMC-based training algorithm for avoiding manually tuning parameters. Based on empirical results, we used the regression task and sampled 10 times more negative items than positive ones for each user out of efficiency concern.
2) **svdfeature** [Chen *et al.*, 2012], also requires negative item sampling but optimizes a ranking-based loss function. It doesn't support MCMC-based training algorithms but only leverages stochastic gradient descent, thus we should manually tune the regularization coefficients and the learning rate. We followed the same sampling strategy as Libfm.
3) **HBF** [Gopalan *et al.*, 2015], a hierarchical Poisson-based matrix factorization, roots from non-negative matrix factorization, but places gamma priors on user/item factors. We used the source code in Github and followed the default settings for parameters.
4) **WARP** [Weston *et al.*, 2010], a ranking-based matrix factorization, tries to optimize precision at the top position. We used a python package developed by Mendeley [Levy and Jack, 2013], where the parameters have been manually tuned.
5) **ICF**, a basis of SBICCF, without any feature provided, will illustrate the effect of features.

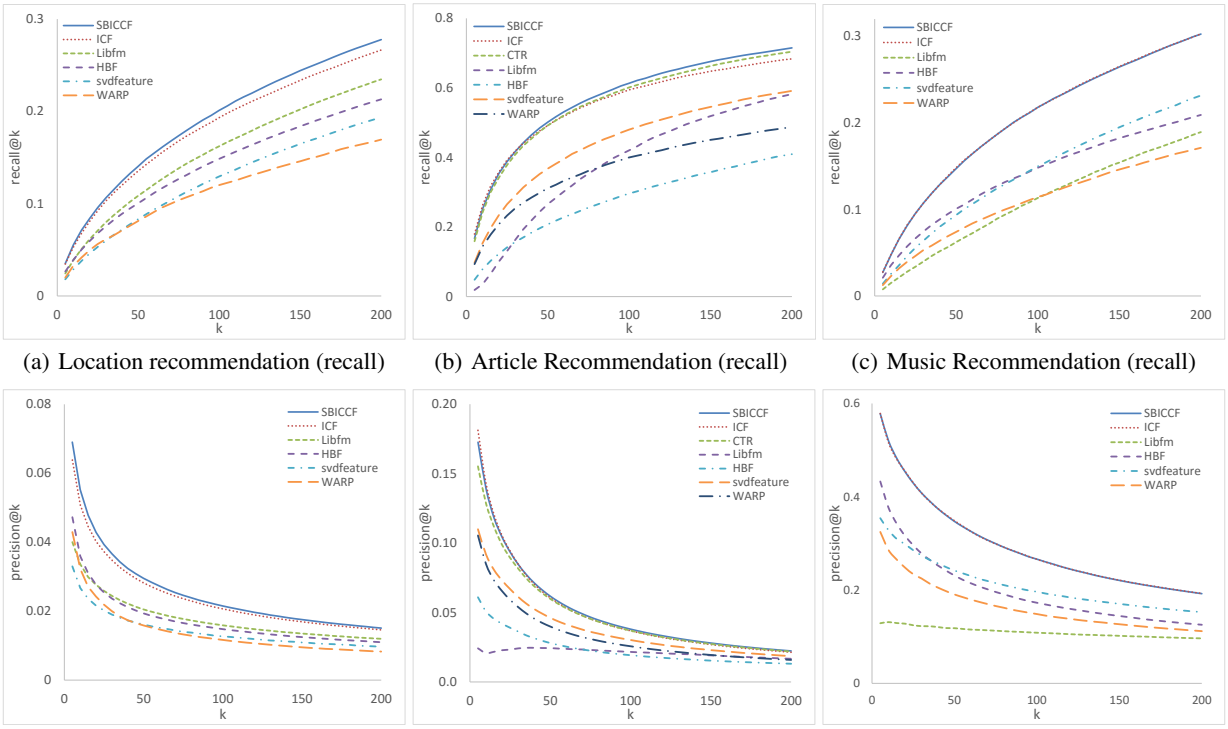
CTR [Wang and Blei, 2011] was also used for the comparison on article recommendation, but it was only designed for combining LDA-based topic modeling with ICF. Thus CTR cannot be considered a baseline of the other two tasks. Since the dataset for article recommendation was the same as CTR, we followed the same settings suggested in the paper. The dimension K of latent space was set to 200 in this case. However, in the other two tasks, K was set to 50.

4.3 Results and Discussions

The overall experimental results of these three tasks are shown in Figure 1. We have the following observations.

First, the proposed algorithm, i.e., SBICCF, outperformed Libfm and svdfeature by a significant margin. This may arise from downweighting the negative preference compared with the positive preference and getting rid of negative sampling. Although svdfeature optimizes a ranking-based loss function and achieves the goal of downweighting negative preference, it doesn't directly target top-k based recommendation metrics. Besides, svdfeature always places the same regularization on the latent factor of items as their features, as shown in Eq (1), so that the regularization of latent factors of item features can not be well controlled.

Second, ICF was significantly better than WARP and HBF. This indicates ICF was still the best choice for collaborative filtering for implicit feedback, at least from the perspectives of the current empirical results. Although WARP directly optimizes the precision at the top position, it exploits the approximation techniques and never converges to the optimal solution. Moreover, it suffers from the computational issues, and becomes more severe with increasing predictive power, just as described in [Li *et al.*, 2015]. Although HBF also downweights the effect of negative preference by an appropriate fit to user activity (i.e., the size of consumed items) so as to capture the diversity of users and the diversity of items, it was only slightly better than the model (i.e., BPF) without downweighting. In other words, their modeling approach didn't show the strong benefit of downweighting, i.e., the varying conference for positive and negative preference,



(a) Location recommendation (recall) (b) Article Recommendation (recall) (c) Music Recommendation (recall)

(d) Location recommendation (precision) (e) Article Recommendation (precision) (f) Music Recommendation (precision)

Figure 1: Comparison among different models on three implicit feedback datasets.

which is slightly different from what has been concluded by [Hu *et al.*, 2008]. However, we can still observe its superiority in some cases to other baselines.

Third, SBICCF was slightly better than CTR. The difference of SBICCF from CTR on article recommendation is that before the addition operation it also maps each topic into the latent space and adds them in the mapped space, i.e., $(\mathbf{t}_j + \mathbf{V}'\boldsymbol{\theta}_j$ v.s. $\mathbf{t}_j + \boldsymbol{\theta}_j)$. The linear mapping matrix $\mathbf{V}$ was learned for aligning topics with items, so that it may learn no more knowledge than the case without aligning, i.e., CTR. However, SPICCF is more general and flexible than CTR since there is no constraint that the number of topics should be the same as the dimension of the latent space.

Last, SBICCF was better than ICF when features were effective. For the tasks of location recommendation and article recommendation, we observed that features played a very important role in improving recommendation, especially for the long tail items. This is not only because SBICCF can determine the relevance automatically but also directly drop irreverent features. For example, in the case of location recommendation, categories such as "my home" and "private place" were determined as irrelevant and directly removed, probably because these categories didn't take any useful information. However, for article recommendation, we found that sparse Bayesian learning in this case almost didn't discover any irrelevant features but only learned the individual contribution of topics. This may be because dimension reduction was performed first so that lots of noises and information with small variances were discarded. For music recommendation, there was no significant improvement, probably due to only a small amount of available user profile information and high density of the Last.fm dataset. When there were sufficient training data for each user, their latent factor can be well learned.

5 Conclusion and Future Work

In this paper, we proposed a sparse Bayesian content-aware collaborative filtering algorithm best tailored to implicit feedback, and developed a scalable optimization algorithm for jointly learning latent factors and hyperparameters. The developed model was not only capable of determining the relevance of each feature but also automatically tuning the regularization coefficients. In theory, we showed that the designed model was equivalent to imposing an adaptive $\ell_{2,1}$ norm on feature latent matrices; and in practical, we observed that it successfully pruned non-informative features. We evaluated the proposed model on three implicit feedback datasets, and showed its consistent superiority to several state-of-the-art content-aware recommendation algorithms.

However, in this framework, each dimension of latent space is assumed independent to each other. Modeling their correlation may lead to some advantages, as shown in [Salakhutdinov and Mnih, 2008], and this goal can be easily achieved based on matrix normal distributions. Although the developed algorithm is capable of pruning non-informative features, it doesn't directly measure the informativeness of each feature. Defining an appropriate metric for the informativeness of features will have significant and practical values, thus it will be studied in a future work.

Acknowledgments

This research was supported by grants from the Natural Science Foundation of China (61502077) and the Fundamental Research Funds for the Central Universities(ZYGX2014Z012, ZYGX2015KYQD018).

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 Trans. Know. Data. Eng.*, 17(6):734–749, 2005.

[Agarwal and Chen, 2009] Deepak Agarwal and Bee-Chung Chen. Regression-based latent factor models. In *Proceedings of KDD'09*, pages 19–28. ACM, 2009.

[Celma, 2010] Oscar Celma. *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer, 2010.

[Chen *et al.*, 2012] Tianqi Chen, Weinan Zhang, Qiuxia Lu, Kailong Chen, Zhao Zheng, and Yong Yu. Svdfeature: a toolkit for feature-based collaborative filtering. *Journal of Machine Learning Research*, 13(1):3619–3622, 2012.

[Gopalan *et al.*, 2014] Prem K Gopalan, Laurent Charlin, and David Blei. Content-based recommendations with poisson factorization. In *Advances in Neural Information Processing Systems*, pages 3176–3184, 2014.

[Gopalan *et al.*, 2015] Prem Gopalan, Jake M Hofman, and David M Blei. Scalable recommendation with hierarchical poisson factorization. In *Proceedings of UAI'15. AUAI Press*, 2015.

[Grippo and Sciandrone, 2000] Luigi Grippo and Marco Sciandrone. On the convergence of the block nonlinear gauss–seidel method under convex constraints. *Operations Research Letters*, 26(3):127–136, 2000.

[Hu *et al.*, 2008] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of ICDM'08*, pages 263–272. IEEE, 2008.

[Koenigstein and Paquet, 2013] Noam Koenigstein and Ulrich Paquet. Xbox movies recommendations: Variational bayes matrix factorization with embedded feature selection. In *Proceedings of RecSys'13*, pages 129–136. ACM, 2013.

[Koren, 2008] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of KDD'08*, pages 426–434. ACM, 2008.

[Levy and Jack, 2013] Mark Levy and Kris Jack. Efficient top-n recommendation by linear regression. In *Large Scale Recommender Systems Workshop in RecSys'13*, 2013.

[Li *et al.*, 2015] Xutao Li, Gao Cong, Xiaoli Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. Rank-geofm: A ranking based geographical factorization method for point of interest recommendation. In *Proceedings of SIGIR'15*, pages 433–442. ACM, 2015.

[Lian *et al.*, 2014] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. Geomf: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *Proceedings of KDD'14*, pages 831–840. ACM, 2014.

[Lian *et al.*, 2015] Defu Lian, Yong Ge, Fuzheng Zhang, Nicholas Jing Yuan, Xing Xie, Tao Zhou, and Yong Rui. Content-aware collaborative filtering for location recommendation based on human mobility data. In *Proceedings of ICDM'15*, pages 261–270. IEEE, 2015.

[Linden *et al.*, 2003] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[Liu *et al.*, 2014] Yong Liu, Wei Wei, Aixin Sun, and Chunyan Miao. Exploiting geographical neighborhood characteristics for location recommendation. In *Proceedings of CIKM'14*, pages 739–748. ACM, 2014.

[MacKay, 1992] David JC MacKay. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

[Pan *et al.*, 2008] R. Pan, Y. Zhou, B. Cao, N.N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *Proceedings of ICDM'08*, pages 502–511. IEEE, 2008.

[Pazzani, 1999] Michael J Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13(5-6):393–408, 1999.

[Rendle *et al.*, 2009] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of UAI'09*, pages 452–461. AUAI Press, 2009.

[Rendle, 2012] Steffen Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57, 2012.

[Salakhutdinov and Mnih, 2008] Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of ICML'08*, pages 880–887. ACM, 2008.

[Stern *et al.*, 2009] David H Stern, Ralf Herbrich, and Thore Graepel. Matchbox: large scale online bayesian recommendations. In *Proceedings of WWW'09*, pages 111–120. ACM, 2009.

[Tipping, 2001] Michael E Tipping. Sparse bayesian learning and the relevance vector machine. *The journal of machine learning research*, 1:211–244, 2001.

[Wang and Blei, 2011] Chong Wang and David M Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of KDD'11*, pages 448–456. ACM, 2011.

[Weston *et al.*, 2010] Jason Weston, Samy Bengio, and Nicolas Usunier. Large scale image annotation: learning to rank with joint word-image embeddings. *Machine learning*, 81(1):21–35, 2010.

[Wipf and Nagarajan, 2008] David P Wipf and Srikantan S Nagarajan. A new view of automatic relevance determination. In *Advances in neural information processing systems*, pages 1625–1632, 2008.

[Wipf and Rao, 2004] David P Wipf and Bhaskar D Rao. Sparse bayesian learning for basis selection. *Signal Processing, IEEE Transactions on*, 52(8):2153–2164, 2004.

[Zhou *et al.*, 2012] Tinghui Zhou, Hanhuai Shan, Arindam Banerjee, and Guillermo Sapiro. Kernelized probabilistic matrix factorization: Exploiting graphs and side information. In *SDM*, volume 12, pages 403–414. SIAM, 2012.