Robust Probabilistic Matrix Factorization

Wu-Jun Li[*]**, Yi Zhen, Dit-Yan Yeung**
CSE Department
HKUST, Hong Kong, China
{liwujun,yzhen,dyyeung}@cse.ust.hk

Zhihua Zhang
College of Computer Science & Technology
Zhejiang University, China
zhzhang@cs.zju.edu.cn

1 Introduction

Recommender systems provide users with personalized suggestions for products or services, which have been widely adopted by electronic commerce sites, such as those of Amazon, Netflix, Google and Yahoo. Most recommender systems are built based on a technique called *collaborative filtering* (CF) which only makes use of the users' past activities such as their transaction history or product satisfaction expressed in ratings, but not the explicit user profiles, to predict the users' future activities.

Probabilistic matrix factorization (PMF) [3] has been shown to achieve state-of-the-art performance for CF. PMF is based on the Gaussian (normal) assumption that both the prior and the likelihood term are assumed to follow normal distributions. However, in real CF applications, it is common to find that some users may deliberately add some rating patterns deviating from the majority just for fun or for some malicious goals. Hence, the rating information from these users might be noninformative or ever harmful for the whole system. These users can be treated as noise (outliers). Furthermore, even for normal users, some of their ratings might not reflect their real preferences due to some unexpected factors. Therefore, noise is inevitable for a real CF system and hence the normal assumption in PMF might not be robust enough for modeling.

The t distribution has been widely used in many applications for modeling cases with noise. In this paper, by adopting the t distribution for both the prior and the likelihood term, a robust probabilistic matrix factorization (RPMF) method is proposed for CF. Furthermore, an expectation-maximization (EM) algorithm [1] is derived to learn the parameters of RPMF. From the EM algorithm, it is easy to see that RPMF is robust against outliers by automatically penalizing the contribution of outliers. Although the nice property of robustness is theoretically guaranteed, we expect to have extensive experimental results, in terms of outlier detection, accuracy and time complexity, to report at the workshop.

[*]Presenter.
Topic: learning algorithms; Preference: Oral/Poster.

2 Robust Probabilistic Matrix Factorization

In a typical CF task, we often use a sparse $N \times M$ matrix $\mathbf{R}$ to represent the rating or preference values on M items given by N users, where each matrix element R_{ij} denotes the preference on item j given by user i. The matrix $\mathbf{R}$ is sparse because many elements are missing, and each such element R_{ij} is assigned the value of 0 to indicate that item j has not been rated by user i. The *goal of CF* is to learn a (real-valued) function to predict (some of) the missing elements in $\mathbf{R}$ based on the observed elements. Let $\mathbf{U} \in \mathbb{R}^{D\times N}$ denote the latent user feature matrix and $\mathbf{V} \in \mathbb{R}^{D\times M}$ denote the latent item feature matrix, with column vectors $\mathbf{U}_{*i}$ and $\mathbf{V}_{*j}$ denoting the user-specific and item-specific latent feature vectors, respectively. The indicator variable Z_{ij} is equal to 1 if user i has rated item j and is 0 otherwise. We use $\mathbf{R}_{i*}$ to denote the actual rating vector by excluding the missing elements, n_i to denote the number of ratings made by user i, i.e., the length of $\mathbf{R}_{i*}$, and $\mathbf{V}^{(i)}$ to denote the columns of $\mathbf{V}$ corresponding to $\mathbf{R}_{i*}$. For example, if user i has only rated items $\{1, 3, 9\}$, then $\mathbf{R}_{i*} = (R_{i1}, R_{i3}, R_{i9})$, $n_i = 3$ and $\mathbf{V}^{(i)} = [\mathbf{V}_{*1}, \mathbf{V}_{*3}, \mathbf{V}_{*9}]$.

2.1 Model Formulation

Given $\mathbf{U}$ and $\mathbf{V}$, the likelihood of the observations is defined as follows:

$$\begin{aligned} p(\mathbf{R}\,|\,\mathbf{U},\mathbf{V}) &= \prod_{i=1}^{N} p(\mathbf{R}_{i*}^T\,|\,\mathbf{U}_{*i},\mathbf{V}) \\ &= \prod_{i=1}^{N} t_\nu(\mathbf{R}_{i*}^T\,|\,{\mathbf{V}^{(i)}}^T\mathbf{U}_{*i}, \lambda\mathbf{I}), \quad (1) \end{aligned}$$

where $t_\nu(\cdot \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the t distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, $\nu > 0$ is the degree of freedom, λ is a parameter, and $\mathbf{I}$ is the identity matrix.

Furthermore, we put a t prior on each $\mathbf{U}_{*i}$ and assume that all $\{\mathbf{U}_{*i}\}$ are independent:

$$p(\mathbf{U}) = \prod_{i=1}^{N} p(\mathbf{U}_{*i}) = \prod_{i=1}^{N} t_\nu(\mathbf{U}_{*i} \,|\, \mathbf{0}, \mathbf{I}). \tag{2}$$

Remark 1 *Unlike PMF, we treat* $\mathbf{V}$ *as parameters rather than missing variables here. We may also put a prior (e.g., Gaussian prior or t prior) on* $\mathbf{V}$ *and apply a variational method [2] to learn* $\mathbf{U}$ *and* $\mathbf{V}$*. This possibility is not considered in this paper though but will be pursued in our future work.*

According to the properties of the t distribution, we can rewrite (1) and (2) as follows:

$$\begin{aligned} p(w_i) &= \mathcal{G}(w_i \,|\, \frac{\nu}{2}, \frac{\nu}{2}), \\ p(\mathbf{U}_{*i} \,|\, w_i) &= \mathcal{N}(\mathbf{U}_{*i} \,|\, \mathbf{0}, \frac{1}{w_i}\mathbf{I}), \qquad (3) \\ p(\mathbf{R}_{i*}^T \,|\, w_i, \mathbf{U}_{*i}) &= \mathcal{N}(\mathbf{R}_{i*}^T \,|\, {\mathbf{V}^{(i)}}^T \mathbf{U}_{*i}, \frac{\lambda}{w_i}\mathbf{I}), \quad (4) \end{aligned}$$

where $\mathcal{G}(\cdot)$ denotes the Gamma distribution, and $\mathcal{N}(\cdot \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

2.2 Learning

We adopt the EM algorithm to learn $\mathbf{U}$ and $\mathbf{V}$. During the learning process, we treat $\boldsymbol{\theta} = (\mathbf{V}, \lambda)$ as parameters and $(\mathbf{U}, \mathbf{w})$ as missing data where $\mathbf{w} = (w_1, w_2, \ldots, w_N)$. The EM algorithm operates by alternating between the E-step and M-step. In the E-step, the expectation of the complete-data log-likelihood is computed, and in the M-step, the parameters $\boldsymbol{\theta} = (\mathbf{V}, \lambda)$ are updated to maximize the expectation of the complete-data log-likelihood. Because all the $\{\mathbf{U}_{*i}\}$ or $\{\mathbf{V}_{*j}\}$ are decoupled during the EM procedure, we can update one column of $\mathbf{U}$ or $\mathbf{V}$ at a time. The EM algorithm is summarized in Algorithm 1.

Compared with PMF, some interesting insights can be observed from Algorithm 1:

- $\bar{w}_i$ can be seen as a weight put on user i. All $\{\mathbf{K}^{(i)}\}$ are computed based on the same values λ and $\mathbf{V}$, which can be seen as the shared covariance information for the ratings of all users.[1] The weight $\bar{w}_i$ will get smaller if $\mathbf{R}_{i*}{\mathbf{K}^{(i)}}^{-1}\mathbf{R}_{i*}^T$ is larger, which means that the ratings of user i can only be *poorly explained* by the shared information λ and $\mathbf{V}$. Subsequently, the outlier users will be automatically penalized.

- ν serves as a smoothing term for updating w_i. The larger the ν, the smaller the penalties will be put on the outlier users. If $\nu \to +\infty$, we can recover PMF by some simple transformation.

- The main difference between RPMF and PMF for updating $\mathbf{V}$ lies in the weights $\{\bar{w}_i\}$ for different users. If some users are outliers, their contribution to the update of $\mathbf{V}$ will be automatically reduced.

[1] Although the $\{\mathbf{K}^{(i)}\}$ of different users may be different, the component $\mathbf{V}_{*j}$ used to compute $\mathbf{K}^{(i)}$ and $\mathbf{K}^{(k)}$ is the same as long as both user i and user k rate item j.

Algorithm 1 EM algorithm for parameter learning in RPMF.

E-step:

$$\begin{aligned} \mathbf{K}^{(i)} &= \lambda\mathbf{I} + {\mathbf{V}^{(i)}}^T\mathbf{V}^{(i)}, \\ \mathbf{H}^{(i)} &= \left(\mathbf{I} + \frac{1}{\lambda}\mathbf{V}^{(i)}{\mathbf{V}^{(i)}}^T\right)^{-1}, \\ \bar{w}_i &= \frac{n_i + \nu}{\mathbf{R}_{i*}{\mathbf{K}^{(i)}}^{-1}\mathbf{R}_{i*}^T + \nu}, \qquad (5) \\ \bar{\mathbf{U}}_{*i} &= \frac{1}{\lambda}\mathbf{H}^{(i)}\mathbf{V}^{(i)}\mathbf{R}_{i*}^T, \qquad (6) \\ \langle w_i\mathbf{U}_{*i}^T\rangle &= \bar{w}_i\bar{\mathbf{U}}_{*i}^T, \\ \langle w_i\mathbf{U}_{*i}\mathbf{U}_{*i}^T\rangle &= \mathbf{H}^{(i)} + \bar{w}_i\bar{\mathbf{U}}_{*i}\bar{\mathbf{U}}_{*i}^T. \end{aligned}$$

M-step:

$$\begin{aligned} \mathbf{V}_{*j} &= \left(\sum_{i=1}^{N} Z_{ij}\langle w_i\mathbf{U}_{*i}\mathbf{U}_{*i}^T\rangle\right)^{-1} \cdot \\ &\quad \left(\sum_{i=1}^{N} Z_{ij}R_{ij}\langle w_i\mathbf{U}_{*i}^T\rangle\right)^T, \qquad (7) \\ \lambda &= \frac{1}{\sum_{i=1}^{N} n_i}\sum_{i=1}^{N}\Big[\bar{w}_i\mathbf{R}_{i*}\mathbf{R}_{i*}^T - 2\langle w_i\mathbf{U}_{*i}^T\rangle\mathbf{V}^{(i)}\mathbf{R}_{i*}^T \\ &\quad + \mathrm{tr}\left(\mathbf{V}^{(i)}{\mathbf{V}^{(i)}}^T\langle w_i\mathbf{U}_{*i}\mathbf{U}_{*i}^T\rangle\right)\Big]. \end{aligned}$$

References

[1] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[2] Y. J. Lim and Y. W. Teh. Variational Bayesian approach to movie rating prediction. In *Proceedings of KDD Cup and Workshop*, 2007.

[3] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS 20*, 2008.