Social Relations Model for Collaborative Filtering

Wu-Jun Li[†] and **Dit-Yan Yeung**[‡]
[†] Shanghai Key Laboratory of Scalable Computing and Systems
Department of Computer Science and Engineering, Shanghai Jiao Tong University, China
[‡] Department of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong, China
liwujun@cs.sjtu.edu.cn, dyyeung@cse.ust.hk

Abstract

We propose a novel probabilistic model for collaborative filtering (CF), called SRMCoFi, which seamlessly integrates both *linear* and *bilinear* random effects into a principled framework. The formulation of SRMCoFi is supported by both social psychological experiments and statistical theories. Not only can many existing CF methods be seen as special cases of SRMCoFi, but it also integrates their advantages while simultaneously overcoming their disadvantages. The solid theoretical foundation of SRMCoFi is further supported by promising empirical results obtained in extensive experiments using real CF data sets on movie ratings.

Introduction

Collaborative Filtering Collaborative filtering (CF) (Breese, Heckerman, and Kadie 1998) does not exploit explicit user profiles but only past activities of the users, such as their transaction history or product satisfaction expressed in ratings, to predict the future activities of the users. In a typical CF task, we often use a sparse matrix R of size $N \times M$ to denote the rating or preference values on the items given by the users. Here N denotes the number of users, M is the number of items, and each matrix element R_{ij} represents the preference of item j given by user i. The matrix R is sparse because many elements are missing, and such elements R_{ij} are 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 function to predict some of the missing elements in R based on the observed elements.

Existing CF methods can be divided into two main categories (Sarwar et al. 2001; Koren 2008; Zhen, Li, and Yeung 2009): memory-based and model-based methods. Memory-based methods predict new ratings by (weighted) averaging the ratings of similar users or items. According to whether similarity is defined based on users or items, memory-based methods can be further divided into user-based methods (Breese, Heckerman, and Kadie 1998) and item-based methods (Sarwar et al. 2001). Unlike memory-based methods, model-based methods learn a model from data using statistical learning techniques. Methods based on matrix factorization (MF) (Takács et al. 2008), or called latent factor models (Koren 2008), are the most representative model-based methods. MF methods can make use of either probabilistic (Lim and Teh 2007; Salakhutdinov and Mnih 2008b) or non-probabilistic (Srebro, Rennie, and Jaakkola 2004; Rennie and Srebro 2005; Takács et al. 2008) formulations. Recently, some hybrid models combining both memory-based and model-based techniques have also emerged (Koren 2008).

Social Relations Model The social relations model[1] (SRM) (Kenny 1994; Li and Loken 2002) has been widely used for modeling dyadic data appeared in many fields, including social psychology, behavioral science and statistical science.[2] SRM was first introduced in the context of *interpersonal perception* studying the *beliefs* that people have about others. During the perception process, there exist a *perceiver* (or called actor) and a *target* (or called partner). For example, Bob might have *beliefs* that his friend, Mary, is intelligent. Here, Bob corresponds to a perceiver and Mary to a target.

To collect dyadic data, there are two basic *designs*: round-robin design and block design. In the *round-robin design*, each person interacts with or rates every other person in a group. In the *block design*, a group is divided into two subgroups and members from each subgroup interact or rate the members of the other subgroup. Sometimes only half of the data are gathered from the block design, which is referred to as a *half-block design*. More specifically, in a half-block design, a group is divided into two subgroups (say A and B), and only the members from one subgroup (say A) rate the members of the other subgroup (say B), but not vice versa. As stated in (Kenny 1994, page 24), a half-block design is typical when the targets are presented on videotapes or movies. Hence, the CF problem can be seen as an example of the half-block design in the perception study because only users rate items but not the other way around.

However, it should be noted that SRM cannot be directly applied to the CF problem because the focus of SRM is on the analysis of variance (ANOVA) over actors, partners as well as the relationships between actors and partners. For

[1] http://davidakenny.net/srm/soremo.htm

[2] http://davidakenny.net/doc/srmbiblio.pdf lists hundreds of references on various applications of SRM.

example, we can use SRM to analyze the actor variance to find out whether people see others as similar in terms of intelligence. Moreover, in general, it is assumed that we can collect all of the data for SRM and hence no elements are missing. For CF, however, the goal is to predict some of the missing ratings, making it fundamentally different from that of the original SRM.

Contributions By adapting SRM, we propose in this paper a novel CF model called SRMCoFi. Some promising properties of SRMCoFi are highlighted here:

- SRMCoFi formulates the CF problem under a probabilistic framework, making it very convenient to learn the model parameters and subsequently control the model capacity automatically.
- SRMCoFi, which subsumes many existing model-based CF methods as special cases, provides a general framework for CF.
- SRMCoFi offers theoretical justifications for many empirical findings by previous CF methods.
- SRMCoFi is very efficient and easily implementable due to the closed-form updates for parameter learning.

Social Relations Model

Let F_{ij} denote the dyadic measurement of the perception or rating from actor i to partner j. In SRM (Kenny 1994), F_{ij} is decomposed into *random effects* a, b, e as follows:

$$F_{ij} = \mu + a_i + b_j + e_{ij} + \epsilon_{ij}, \tag{1}$$

where μ is a constant representing the overall mean, a_i is the *actor effect* representing the average level of rating of an actor in the presence of a variety of partners, b_j is the *partner effect* representing the average level of a rating which a person elicits from a variety of actors, e_{ij} is the *relationship effect* representing the rating of an actor toward a particular partner, above and beyond their actor and partner effects, and ϵ_{ij} is an error term representing the unstable aspect of the perception. In particular, for the movie rating application, actors refer to users and partners to movies.

In general, a, b, e, ϵ are assumed to be random variables with the following distributions:

$$\begin{aligned} a_i &\overset{iid}{\sim} \mathcal{N}(\cdot|0, \sigma_a^2), & b_j &\overset{iid}{\sim} \mathcal{N}(\cdot|0, \sigma_b^2), \\ e_{ij} &\sim \mathcal{N}(\cdot|0, \sigma_e^2), & \epsilon_{ij} &\overset{iid}{\sim} \mathcal{N}(\cdot|0, \sigma_\epsilon^2), \\ \mathrm{Cov}(a_i, b_i) &\neq 0, & \mathrm{Cov}(e_{ij}, e_{ji}) &\neq 0, \end{aligned} \tag{2}$$

where $\mathrm{Cov}()$ denotes the covariance. In SRM, all the other covariances are assumed to be 0. $\mathrm{Cov}(a_i, b_i) \neq 0$ means that the actor effect and partner effect of the *same* individual i have nonzero covariance. Since CF is a half-block design, one specific individual can only have either actor effect or partner effect, but not both. Hence, it is unnecessary to consider these covariances for CF.

The focus of SRM is not on estimating the effects for specific individuals and relationships but on estimating the variance due to effects. So, as said above, ANOVA (Li and Loken 2002) is the main goal of the original SRM. Moreover, SRM typically assumes that all of the data are observed. This is completely different from the goal of CF which is to predict some of the missing elements in the rating matrix. Hence, SRM cannot be directly applied to CF.

SRMCoFi

Since CF is fundamentally different from traditional applications of SRM, we need to adapt SRM appropriately if we want to apply it to CF. The result is our social relations model for collaborative filtering, which we abbreviate as SRMCoFi in the sequel.

Model

SRMCoFi defines the likelihood function over the observed ratings as follows:

$$\begin{aligned} F_{ij} &= \mu + a_i + b_j + U_i^T V_j, \\ E(R_{ij}|F_{ij}) &= g^{-1}(F_{ij}), \\ p(R|F, \phi) &= \prod_{i=1}^{N} \prod_{j=1}^{M} \left[p(R_{ij}|F_{ij}, \phi)\right]^{I_{ij}}, \end{aligned} \tag{3}$$

where $U \in \mathbb{R}^{D \times N}$ is the latent user feature matrix and $V \in \mathbb{R}^{D \times M}$ is the latent movie feature matrix, with column vectors U_i and V_j denoting user-specific and movie-specific latent feature vectors respectively, $E(\cdot)$ is the expectation function, $g(\cdot)$ is the link function in a generalized linear model (GLM) (Gelman et al. 2003), ϕ is the variance or *dispersion parameter* of the GLM (Gelman et al. 2003), and I_{ij} is an indicator variable which is equal to 1 if user i rated movie j and 0 otherwise. For example, if $g^{-1}(F_{ij}) = \exp(F_{ij})$ and $p(R_{ij}|F_{ij}) = \mathrm{Poisson}(R_{ij}|g^{-1}(F_{ij}))$, the model is equivalent to a Poisson regression model (Gelman et al. 2003) with the log-link function, which is always used to model data taking the form of counts. One representative application of the Poisson regression model is epidemiology where the incidence of diseases is studied (Gelman et al. 2003, page 51).

The SRMCoFi model defined in (3) adapts SRM in two aspects:

- SRMCoFi adopts a *bilinear* term $U_i^T V_j$ to represent the relationship effect e_{ij} of SRM. Hence, SRMCoFi seamlessly integrates the *linear* (a_i, b_j) and *bilinear* effects into a principled framework. This adaptation is very crucial for SRMCoFi to achieve the goal of CF, which is to predict some missing elements in the preference matrix. After learning the model, each user i is associated with a latent feature vector U_i and each movie j with a latent feature vector V_j. Based on these learned feature vectors, the missing elements in the preference matrix can be predicted easily. For example, the expected value $E(R_{ij}|F_{ij})$ computed based on (3) can be used as the predicted value for a missing element R_{ij}, which is the strategy used in this paper. On the other hand, it is not easy to directly apply SRM to predict the missing elements.
- By exploiting different GLMs, SRMCoFi generalizes SRM to accommodate different types of data, such as binary measurements and counting data.

We gave an example above on how to model counting data via a Poisson regression model based on the log-link function. Here we give two other examples among many possibilities:

- Logistic regression model: If R_{ij} is binary, we can define $g^{-1}(F_{ij}) = \frac{\exp(F_{ij})}{1+\exp(F_{ij})}$ and $p(R_{ij}|F_{ij}) = \text{Bernoulli}(R_{ij}|g^{-1}(F_{ij}))$. Here, the link is the logit function and there is no dispersion parameter ϕ. As for CF, if the task is to predict whether user i has rated movie j, i.e., to predict I_{ij}, this model might be a good choice.

- Normal-linear model: If $g^{-1}(F_{ij}) = F_{ij}$ and $p(R_{ij}|F_{ij},\phi) = \mathcal{N}(R_{ij}|F_{ij},\sigma^2)$, it is a normal-linear model with $\phi = \sigma^2$, which is a typical choice for modeling continuous values.

Other combinations of $g^{-1}(F_{ij})$ and $p(R_{ij}|F_{ij},\phi)$ can be used to devise different models for different data types.[3] Hence, SRMCoFi provides a flexible framework for modeling data from diverse applications.

In this paper, we only study the normal-linear model in detail with $g^{-1}(F_{ij}) = F_{ij}$ and $p(R_{ij}|F_{ij},\phi) = \mathcal{N}(R_{ij}|F_{ij},\sigma^2)$. The learning and inference procedures for other variants are similar except that the closed-form solutions for the normal-linear model may not exist for other variants. Nevertheless, even if this is the case, we can still resort to a gradient method (Shewchuk 1994) to obtain solutions. Due to the page limit, such variants will not be described in detail in this paper.

We place Gaussian priors on the latent variables of SRMCoFi:

$$p(\mathbf{a}) = \prod_{i=1}^{N} \mathcal{N}(a_i|0,\sigma_a^2), \quad p(\mathbf{b}) = \prod_{j=1}^{M} \mathcal{N}(b_j|0,\sigma_b^2),$$
$$p(U) = \prod_{i=1}^{N} \mathcal{N}(U_i|0,\sigma_U^2\mathbf{I}), \quad p(V) = \prod_{j=1}^{M} \mathcal{N}(V_j|0,\sigma_V^2\mathbf{I}),$$
$$p(\mu) = \mathcal{N}(\mu|0,\sigma_\mu^2), \tag{4}$$

where $\mathbf{a} = (a_1, a_2, \ldots, a_N)^T$, $\mathbf{b} = (b_1, b_2, \ldots, b_M)^T$, and $\mathbf{I}$ is the identity matrix. Here, the latent variables $\mu, \mathbf{a}, \mathbf{b}, U, V$ are referred to as *parameters* and $\sigma^2, \sigma_\mu^2, \sigma_a^2, \sigma_b^2, \sigma_U^2, \sigma_V^2$ as *hyperparameters*.

We assume independence among the parameters and hence $p(\mu, \mathbf{a}, \mathbf{b}, U, V) = p(\mu)p(\mathbf{a})p(\mathbf{b})p(U)p(V)$. Furthermore, the likelihood of the normal-linear model is: $p(R|\mu, \mathbf{a}, \mathbf{b}, U, V, \sigma^2) = p(R|F, \sigma^2) = \prod_{i=1}^{N}\prod_{j=1}^{M} \left[\mathcal{N}(R_{ij}|F_{ij},\sigma^2)\right]^{I_{ij}}$.

Thus, the log of the posterior distribution over the *parameters* can be written as follows:

$$\ln p(\mu, \mathbf{a}, \mathbf{b}, U, V|R, \sigma^2, \sigma_\mu^2, \sigma_a^2, \sigma_b^2, \sigma_U^2, \sigma_V^2) =$$
$$-\frac{1}{2\sigma^2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - F_{ij})^2 - \frac{1}{2\sigma_U^2}\sum_{i=1}^{N} U_i^T U_i$$
$$-\frac{1}{2\sigma_V^2}\sum_{j=1}^{M} V_j^T V_j - \frac{1}{2\sigma_\mu^2}\mu^2 - \frac{1}{2\sigma_a^2}\sum_{i=1}^{N} a_i^2$$
$$-\frac{1}{2\sigma_b^2}\sum_{j=1}^{M} b_j^2 - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}\ln\sigma^2 - \frac{ND}{2}\ln\sigma_U^2$$
$$-\frac{MD}{2}\ln\sigma_V^2 - \frac{1}{2}\ln\sigma_\mu^2 - \frac{N}{2}\ln\sigma_a^2 - \frac{M}{2}\ln\sigma_b^2 + C, \tag{5}$$

where C is a constant that does not depend on the parameters and hyperparameters.

For fixed hyperparameters, maximizing the log-posterior over the parameters is equivalent to minimizing the following objective function:

$$\mathcal{L} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - F_{ij})^2 + \frac{\lambda_U}{2}\text{tr}(U^T U)$$
$$+\frac{\lambda_V}{2}\text{tr}(V^T V) + \frac{\lambda_\mu}{2}\mu^2 + \frac{\lambda_a}{2}\mathbf{a}^T\mathbf{a} + \frac{\lambda_b}{2}\mathbf{b}^T\mathbf{b}, \tag{6}$$

where $\text{tr}(\cdot)$ denotes the trace of a matrix, $\lambda_U = \sigma^2/\sigma_U^2$, $\lambda_V = \sigma^2/\sigma_V^2$, $\lambda_\mu = \sigma^2/\sigma_\mu^2$, $\lambda_a = \sigma^2/\sigma_a^2$, and $\lambda_b = \sigma^2/\sigma_b^2$.

Parameter Learning

The learning procedure for SRMCoFi consists of two parts: learning the parameters and learning the hyperparameters. We will discuss hyperparameter learning in detail in the following subsection. In this subsection, we only consider the parameter learning problem, which is to optimize the objective function in (6) with fixed hyperparameters (or, equivalently, with fixed λ's).[4]

Note that (6) is not jointly convex over $\mu, \mathbf{a}, \mathbf{b}, U, V$. However, it is convex with respect to each individual variable with the others held fixed. We exploit this property to perform alternating updates for each variable iteratively, the convergence of which can be guaranteed and will be proved later in this subsection.

To update μ, we compute the gradient of $\mathcal{L}$ with respect to μ and set the gradient to 0, giving

$$\mu = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}\left[R_{ij} - (a_i + b_j + U_i^T V_j)\right]}{\lambda_\mu + \sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}}. \tag{7}$$

Similarly, closed-form solutions exist for updating a_i, b_j:

$$a_i = \frac{\sum_{j=1}^{M} I_{ij}(R_{ij} - \mu - b_j - U_i^T V_j)}{\lambda_a + \sum_{j=1}^{M} I_{ij}}, \tag{8}$$

$$b_j = \frac{\sum_{i=1}^{N} I_{ij}(R_{ij} - \mu - a_i - U_i^T V_j)}{\lambda_b + \sum_{i=1}^{N} I_{ij}}. \tag{9}$$

[3] Theoretically any combination may be adopted. By taking into consideration the computational requirements, however, only some of these combinations are good choices for real applications. For example, we always constrain $g^{-1}(F_{ij})$ and $p(R_{ij}|F_{ij},\phi)$ to what can give a concave log-likelihood function.

[4] Here we overload the term 'hyperparameters' to refer to the λ's. We can easily realize from the context whether 'hyperparameters' refer to the σ^2s or the λ's. In fact, they refer to the same thing because the λ's can be computed from the σ^2s.

To update U, we first rewrite (6) as follows:

$$\mathcal{L} = \frac{1}{2}\sum_{i=1}^{N}\left\{U_i^T\left(\lambda_U\mathbf{I}+\sum_{j=1}^{M}I_{ij}V_jV_j^T\right)U_i \right. \\ \left. -2\left[\sum_{j=1}^{M}I_{ij}(R_{ij}-\mu-a_i-b_j)V_j^T\right]U_i\right\}+C_1, \quad (10)$$

where C_1 is a constant independent of U. We can see that the vectors U_i for different i are decoupled, which means that we can solve a linear system of equations for each row of U separately.

It is not difficult to see that the objective function in (10) is convex. If D is small, we can directly solve the linear system to obtain: $U_i = \left(\lambda_U\mathbf{I}+\sum_{j=1}^{M}I_{ij}V_jV_j^T\right)^{-1}\left[\sum_{j=1}^{M}I_{ij}(R_{ij}-\mu-a_i-b_j)V_j\right]$. Otherwise, if D is large, we can instead apply the steepest descent method (Shewchuk 1994) with convergence guaranteed.

Similarly, updating V corresponds to solving the following linear system:

$$\left(\lambda_V\mathbf{I}+\sum_{i=1}^{N}I_{ij}U_iU_i^T\right)V_j = \sum_{i=1}^{N}I_{ij}(R_{ij}-\mu-a_i-b_j)U_i.$$

Theorem 1 *The learning procedure above guarantees local convergence.*

Proof: (Sketch) The objective function in (6) is lower bounded by 0. Furthermore, the objective function decreases its value monotonically in each iteration when the variables are updated. Hence, the learning procedure always converges to a local minimum. □

Theorem 2 *Both the time complexity and space complexity are linear in the number of training ratings available in R.*

Proof: Let us denote the number of training ratings by γ. It is easy to see that the time needed for updating $\mu, \mathbf{a}, \mathbf{b}$ is $O(D\gamma)$. For updating U, the time for computing all the $\sum_{j=1}^{M}(\cdot)$ for all U_i is $O(D\gamma)$, and for each U_i, $O(D^3+D^2)$ is needed for computing the matrix inverse and multiplication. Hence, the total time for updating U is $O(D\gamma+D^3N)$. Similarly, $O(D\gamma+D^3M)$ is needed for updating V. Hence, the time complexity for each iteration is $O(\gamma)$ because D is typically a small constant and $N, M \ll \gamma$. Moreover, from our experiment, only a small number (< 20) of iterations is sufficient for the algorithm to achieve good performance. Hence, we can say that the time complexity of the whole algorithm is $O(\gamma)$.

The space complexity of $R, \mu, \mathbf{a}, \mathbf{b}, U, V$ is $O(\gamma), O(1), O(N), O(M), O(DN), O(DM)$ respectively. Hence, the overall space complexity is also $O(\gamma)$. □

Hyperparameter Learning

How to choose appropriate regularization parameters (or called hyperparameters here), such as the λ's in (6), is crucial to model capacity control. For models based on non-probabilistic formulations, a validation set or, more generally, cross validation is often used to determine the hyperparameters. However, in cross validation, only a small number of discrete values are considered as candidate choices. Hence, the values eventually chosen for the hyperparameters are not necessarily optimal.

Probabilistic formulation allows the hyperparameters to be learned automatically. In this paper, we adopt an alternating method to efficiently learn the hyperparameters. More specifically, in the learning procedure, we fix the hyperparameters and optimize over the parameters first and then optimize over the hyperparameters with the parameters kept fixed, and this procedure iterates until some termination condition is satisfied. In this paper, we use a conjugate hyperprior which gives closed-form updates for inference. More specifically, we use an inverse-χ^2 distribution as a conjugate hyperprior for the hyperparameters (σ^2s).

Assume that $p(y|0,\sigma_y^2) = \mathcal{N}(y|0,\sigma_y^2)$ with σ_y^2 unknown, the likelihood of n i.i.d. observations is

$$p(\mathbf{y}|\sigma_y^2) \propto \frac{1}{\sigma_y^n}\exp\left(-\frac{\sum_{i=1}^{n}y_i^2}{2\sigma_y^2}\right) = \frac{1}{\sigma_y^n}\exp\left(-\frac{n}{2\sigma_y^2}s\right),$$

where s is the sufficient statistic: $s(\sigma_y^2) = \frac{\sum_{i=1}^{n}y_i^2}{n}$.

If we use a scaled inverse-χ^2 (Gelman et al. 2003) as the hyperprior for σ_y^2: $\sigma_y^2 \sim \text{Inv-}\chi^2(\nu_0,\sigma_0^2)$, we can get

$$\sigma_y^2|\mathbf{y} \sim \text{Inv-}\chi^2\left(\nu_0+n, \frac{\nu_0\sigma_0^2+ns}{\nu_0+n}\right). \quad (11)$$

Hence, the hyperprior can be thought of as providing prior information equivalent to ν_0 observations with average squared deviation σ_0^2 (Gelman et al. 2003).

It is easy to derive the sufficient statistics of the σ^2s for SRMCoFi:

$$s(\sigma^2) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}I_{ij}(R_{ij}-F_{ij})^2}{\sum_{i=1}^{N}\sum_{j=1}^{M}I_{ij}}, \quad s(\sigma_\mu^2) = \mu^2,$$

$$s(\sigma_a^2) = \frac{\sum_{i=1}^{N}a_i^2}{N}, \quad s(\sigma_b^2) = \frac{\sum_{j=1}^{M}b_j^2}{M},$$

$$s(\sigma_U^2) = \frac{\sum_{i=1}^{N}U_i^TU_i}{ND}, \quad s(\sigma_V^2) = \frac{\sum_{j=1}^{M}V_j^TV_j}{MD}.$$

The mode, which is $\frac{\nu}{\nu+2}\sigma_y^2$ for Inv-$\chi^2(\nu,\sigma_y^2)$, of the posterior of the σ^2 is adopted to update the λ's in (6). It is easy to see that the time and space complexity of hyperparameter learning is also linear in the number of training ratings.

Comparison with Related Work

Let us first consider MF methods based on non-probabilistic formulations. Maximum margin matrix factorization (MMMF) (Srebro, Rennie, and Jaakkola 2004; Rennie and Srebro 2005) can be seen as a special case of (6) by setting $\mu = a_i = b_j = \lambda_\mu = \lambda_a = \lambda_b = 0$.[5]

[5]The original MMMF (Srebro, Rennie, and Jaakkola 2004) only used the hinge loss, but many subsequent works also used other loss functions such as smooth hinge (Rennie and Srebro 2005) and squared loss (Weimer, Karatzoglou, and Smola 2008). Here we refer to the squared loss function.

It has been demonstrated empirically by many researchers (Takács et al. 2008; Weimer, Karatzoglou, and Smola 2008) that adding offset (or bias) terms to MMMF can improve its performance. These methods, MMMF+offset in (Weimer, Karatzoglou, and Smola 2008) and BRISMF in (Takács et al. 2008), are equivalent to (6) by setting $\mu = \lambda_\mu = 0$. All these non-probabilistic methods have shortcomings when it comes to choosing the hyperparameters.

Other MF methods (Lim and Teh 2007; Salakhutdinov and Mnih 2008a; 2008b) are based on probabilistic formulations. PMF (Salakhutdinov and Mnih 2008b) can be seen as a special case of SRMCoFi by removing the terms for $\mu, \mathbf{a}, \mathbf{b}$. Bayesian PMF (BPMF) (Salakhutdinov and Mnih 2008a) extends PMF by putting some hyperpriors on the hyperparameters and uses Gibbs sampling for learning and inference. The underlying formulation of VB (Lim and Teh 2007) is the same as Bayesian PMF except that VB adopts variational methods for Bayesian inference. Although SRMCoFi in this paper does not perform fully Bayesian inference, the techniques used in extending PMF to Bayesian PMF (or MMMF to VB) may also be applied here to extend SRMCoFi to its fully Bayesian counterpart, possibly incurring higher computation cost to take advantage of a fully Bayesian approach. In that case, BPMF and VB can be seen as special cases of the corresponding fully Bayesian counterparts of SRMCoFi. We will pursue this direction in our future work. One common characteristic of these probabilistic formulations is that they do not include the very important *linear terms* used in SRMCoFi. Actually, many variants of MF methods were empirically studied in (Takács et al. 2008) and MMMF with the added *linear terms* (called BRISMF) was found to perform the best. This shows that the importance of the *linear terms* as supported by psychological experiments is also verified by empirical findings in CF applications.

Modeling the offset terms (i.e., the linear random effects in SRMCoFi) and hyperparameter learning are two important characteristics for a promising CF model. However, existing methods do not simultaneously take both of them into consideration. Although SRMCoFi has been motivated by the study of social relations, it coincides with a model which seamlessly integrates these two aspects into a principled framework. The solid foundation validated by theories and experiments from psychological and statistical studies provides justification for the inclusion of the offset terms. Hence, compared with existing methods, better performance can be expected from SRMCoFi.

Experiments

Data Sets and Evaluation Metric

We evaluate SRMCoFi on two widely used data sets: MovieLens (Sarwar et al. 2001) and EachMovie (Breese, Heckerman, and Kadie 1998). Some information about the data sets is summarized in Table 1.

Root mean squared error (RMSE) is used as the *error measure*. Suppose there are totally H elements to be predicted, and R_h and $\tilde{R}_h$ denote the ground-truth and predicted values respectively for element h. Then the RMSE is defined as follows: $\mathrm{RMSE} = \sqrt{\frac{\sum_{h=1}^{H}(R_h - \tilde{R}_h)^2}{H}}$. We can see that the smaller the RMSE is, the better the method will be.

Table 1: Data sets for evaluation.

Dataset	# Users	# Movies	# Ratings
MovieLens	943	1,682	100,000
EachMovie	61,265	1,623	2,811,718

Effect of Hyperparameter Learning

Here, we compare SRMCoFi with MMMF+offset (Weimer, Karatzoglou, and Smola 2008). We keep part of the training data as a validation set to determine the hyperparameters for MMMF+offset, while the hyperparameters of SRMCoFi are automatically learned with our proposed algorithm. Hence, the main difference between MMMF+offset and SRMCoFi lies in how the hyperparameters are determined.

We randomly select 20%, 40%, 60% and 80% of the data to form the training set and use the rest as the test set for evaluation. This random selection is carried out 10 rounds independently for each splitting ratio. In all the experiments in this paper, we set $D = 30$. The mean and standard deviation of the RMSE are reported in Table 2. Two other approaches, UserMean and MovieMean, are also included for comparison. UserMean uses the sample mean of the same user's training ratings for prediction, and MovieMean uses that of the same movie's training ratings. From the results, we can see that hyperparameter learning is very important for CF and SRMCoFi is very effective.

Table 2: Comparison between SRMCoFi and MMMF+offset to show that hyperparameter learning is effective. The best performance (with smallest RMSE) is shown in bold. All standard deviations are 0.002 or less.

		20%	40%	60%	80%
MovieLens	UserMean	1.060	1.049	1.044	1.043
	MovieMean	1.054	1.036	1.030	1.027
	MMMF+offset	0.974	0.954	0.945	0.940
	SRMCoFi	**0.969**	**0.941**	**0.918**	**0.910**
EachMovie	UserMean	1.477	1.443	1.432	1.427
	MovieMean	1.389	1.388	1.387	1.387
	MMMF+offset	1.286	1.233	1.191	1.161
	SRMCoFi	**1.245**	**1.195**	**1.160**	**1.136**

Effect of Linear Random Effects

MMMF+offset Vs MMMF The difference between MMMF+offset and MMMF lies in the linear offset terms. Here, we compare them to demonstrate the significance of these terms. Both MMMF+offset and MMMF use a validation set to determine the hyperparameters, with the same data set partitioning scheme as that above. The results are shown in Table 3, from which we can see that the linear offset terms dramatically improve performance.

Table 3: Comparison between MMMF+offset and MMMF to show the effectiveness of the offset terms. All standard deviations are 0.002 or less.

		20%	40%	60%	80%
MovieLens	MMMF	1.040	0.981	0.964	0.955
	MMMF+offset	**0.974**	**0.954**	**0.945**	**0.940**
EachMovie	MMMF	1.377	1.281	1.265	1.208
	MMMF+offset	**1.286**	**1.233**	**1.191**	**1.161**

Comparison with PMF Both PMF (Salakhutdinov and Mnih 2008b) and SRMCoFi can perform hyperparameter learning automatically. Hence, the only difference between them lies in the linear terms. The results are shown in Table 4, which once again verifies that the linear terms are very crucial for CF.

Table 4: Comparison between SRMCoFi and PMF. All standard deviations are 0.002 or less.

		20%	40%	60%	80%
MovieLens	PMF	1.029	0.977	0.956	0.952
	SRMCoFi	**0.969**	**0.941**	**0.918**	**0.910**
EachMovie	PMF	1.324	1.233	1.224	1.178
	SRMCoFi	**1.245**	**1.195**	**1.160**	**1.136**

Comparison with VB and MVTM We further compare SRMCoFi with other state-of-the-art methods, including VB (Lim and Teh 2007) and the *matrix-variate t model* (MVTM) (Zhu, Yu, and Gong 2008). As in (Zhu, Yu, and Gong 2008), we randomly select 80% of the EachMovie data for training and use the rest for testing. The mean and standard deviation over 10 rounds are reported in Table 5. The better performance of SRMCoFi can be directly attributed to the linear random effects of SRMCoFi because VB only has bilinear terms.

Table 5: Comparison of SRMCoFi with VB and MVTM. All standard deviations are 0.002 or less.

VB	MVTM (mode)	MVTM (mean)	SRMCoFi
1.165	1.162	1.151	**1.136**

Computational Cost

In real implementation, we find that SRMCoFi is quite efficient. In the experiment on EachMovie with 80% of data for training, it takes less than 10 minutes to complete one iteration of both hyperparameter learning and parameter learning, which is comparable with a C++ implementation of $CoFi^{Rank}$ (Weimer et al. 2008) even though our SRMCoFi implementation is in MATLAB. Compared with the MMMF implementation in (Rennie and Srebro 2005), SRMCoFi is an order of magnitude faster.

Conclusion

SRMCoFi, which has been inspired by social psychological studies, can be seen as a model that seamlessly integrates both *hyperparameter learning* and the use of *offset terms* into a principled framework. As a result, SRMCoFi integrates the advantages of existing methods and simultaneously overcomes their disadvantages, achieving very promising performance in real CF tasks. To a certain extent, a major contribution of SRMCoFi is that it serves to provide justifications from social psychological studies to many empirical findings by existing CF methods.

Acknowledgments

Li is supported by a grant from the "project of Arts & Science" of Shanghai Jiao Tong University (No. 10JCY09). Yeung is supported by General Research Fund 621310 from the Research Grants Council of Hong Kong.

References

Breese, J. S.; Heckerman, D.; and Kadie, C. M. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *UAI*, 43–52.

Gelman, A.; Carlin, J. B.; Stern, H. S.; and Rubin, D. B. 2003. *Bayesian Data Analysis, Second Edition*. Chapman & Hall/CRC.

Kenny, D. A. 1994. *Interpersonal Perception: A Social Relations Analysis*. Guilford Publications, Inc., New York.

Koren, Y. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD*, 426–434.

Li, H., and Loken, E. 2002. A unified theory of statistical analysis and inference for variance component models for dyadic data. *Statistica Sinica* 12:519–535.

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

Rennie, J. D. M., and Srebro, N. 2005. Fast maximum margin matrix factorization for collaborative prediction. In *ICML*, 713–719.

Salakhutdinov, R., and Mnih, A. 2008a. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *ICML*, 880–887.

Salakhutdinov, R., and Mnih, A. 2008b. Probabilistic matrix factorization. In *NIPS 20*.

Sarwar, B. M.; Karypis, G.; Konstan, J. A.; and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW*, 285–295.

Shewchuk, J. R. 1994. An introduction to the conjugate gradient method without the agonizing pain. Technical report, Pittsburgh, PA, USA.

Srebro, N.; Rennie, J. D. M.; and Jaakkola, T. 2004. Maximum-margin matrix factorization. In *NIPS*.

Takács, G.; Pilászy, I.; Németh, B.; and Tikk, D. 2008. Investigation of various matrix factorization methods for large recommender systems. In *Proc. of the 2nd KDD Workshop on Large Scale Recommender Systems and the Netflix Prize Competition*.

Weimer, M.; Karatzoglou, A.; Le, Q.; and Smola, A. 2008. Cofi rank - maximum margin matrix factorization for collaborative ranking. In *NIPS 20*.

Weimer, M.; Karatzoglou, A.; and Smola, A. 2008. Improving maximum margin matrix factorization. *Machine Learning* 72(3):263–276.

Zhen, Y.; Li, W.-J.; and Yeung, D.-Y. 2009. TagiCoFi: tag informed collaborative filtering. In *RecSys*, 69–76.

Zhu, S.; Yu, K.; and Gong, Y. 2008. Predictive matrix-variate t models. In *NIPS 20*.