A Bayesian Treatment of Social Links in Recommender Systems

Mike Gartrell, Ulrich Paquet, and Ralf Herbrich

Technical Report
CU-CS-1092-12
May 2012

University of Colorado Boulder
Department of Computer Science
430 UCB
Boulder, Colorado 80309-0430
www.cs.colorado.edu

A Bayesian Treatment of Social Links in Recommender Systems

Mike Gartrell
Department of Computer Science
University of Colorado Boulder
Boulder, CO
mike.gartrell@colorado.edu

Ulrich Paquet
Microsoft Research Cambridge
Cambridge, UK
ulripa@microsoft.com

Ralf Herbrich
Facebook
Menlo Park, CA
ralf@fb.com

ABSTRACT

Recommender systems are increasingly driving user experiences on the Internet. This personalization is often achieved through the factorization of a large but sparse observation matrix of user-item feedback signals. In instances where the user's social network is known, its inclusion can significantly improve recommendations for cold start users. There are numerous ways in which the network can be incorporated into a probabilistic graphical model. We propose and investigate two ways for including a social network, either as a Markov Random Field that describes a user similarity in the prior over user features, or an explicit model that treats social links as observations. State of the art performance is reported on the Flixster online social network dataset.

1. INTRODUCTION AND RELATED WORK

The Web has experienced explosive growth over the past decade. Concurrent with the growth of the Web, recommender systems have attracted increasing attention. Recommender systems aid users in selecting content that is most relevant to their interests, and notable examples of popular recommender systems are available for a variety of types of online content, including movies [10], books [1], music [11], and news [7].

Online social networks (OSNs), such as Facebook [2], Google+ [6], and LinkedIn [8], have quickly become the fastest growing part of the Web. For example, Facebook has grown dramatically over the past three years, from 100 million users in August 2008 [3] to 800 million users as of September 2011 [4]. This rapid growth in OSNs presents a substantial opportunity for recommender systems that are able to effectively leverage OSN data for providing recommendations.

The task of a recommender system is to predict which items will be of interest to a particular user. Recommender systems are generally implemented using one of two approaches: content filtering and collaborative filtering. The content filtering approach builds profiles that describe both users and items. For example, users may be described by demographic information such as age and gender, and items may be described by attributes such as genre, manufacturer, and author. One popular example of content filtering is the Music Genome Project [9] used by Pandora [11] to recommend music.

Collaborative filtering is an alternative to content filtering, and relies only on past user behavior without using explicit user and item profiles. Examples of past user behavior include previous transactions, such as a user's purchase history, and users' ratings on items. Collaborative filtering learns about users and items based on the items that users have rated and users that have rated similar items. A major appeal of collaborative filtering systems is that they do not require the creation of user and item profiles, which require obtaining external information that may not be easy to collect. As such, collaborative filtering systems can be easily applied to a variety of domains, such as movies, music, etc.

There are two primary approaches to collaborative filtering: neighborhood methods and latent factor models. Neighborhood method involve computing relationships between items or between users. Item-based neighborhood approaches [13, 20, 29] predict a user's rating for an item based on ratings of similar items rated by the same user. User-based neighborhood approaches [12, 18] predict a user's rating for an item based on the ratings of similar users on the item. Item-based and user-based approaches generally use a similarity computation algorithm to compute a neighborhood of similar items or users; examples of similarity algorithms include the Pearson Correlation Coefficient algorithm and the Vector Space Similarity algorithm.

In contrast to neighborhood methods, latent factor models use an alternative approach that characterizes users and items in terms of factors inferred from patterns in ratings data. In the case of movies, the inferred factors might be a measure of traits such as genre aspects (e.g., horror vs. comedy), the extent to which a movie is appealing to females, etc. For users, each factor indicates the extent to which a user likes items that have high scores on the corresponding item factors.

Some of the most successful recommender systems that use latent factor models are based on matrix factorization approaches [26, 27, 28, 30]. As described in Section 3, matrix factorization models learn a mapping of users and items to a join latent feature/factor trait space of dimensionality K. User-item interactions are modeled as inner products in this trait space. The inner product between each user and item feature vector captures the user's overall interest in the item's traits.

Traditionally, most recommender systems have not considered the relationships between users in social networks. More recently, however, a number of approaches to social-based recommender systems have been proposed and evaluated. Most OSN-based approaches assume a social network among users and make recommendations for a user based on the ratings of users that have social connections to the specified user.

Several neighborhood-based approaches to recommendation in OSNs have been proposed [24, 16, 15, 32]. These approaches generally explore the social network and compute a neighborhood of users trusted by a specified user. Using this neighbor, these systems provide recommendations by

aggregating the ratings of users in this trust neighborhood. Since these systems require exploration of the social network, these approaches tend to be slower than social-based latent factor models when computing predictions.

Some latent factor models for social-based recommendation have also been proposed [21, 22, 23, 17, 31]. These methods use matrix factorization to learn latent features for user and items from the observed ratings and from users' friends (neighbors) in the social network. Experimental results show better performance than neighborhood-based approaches.

In this work we present two matrix factorization models for recommendation in social networks. We represent each user and item by a vector of features. We model the social network as a undirected graph with binary friendship links between users. Such a model is the common case for most OSNs. Our work makes the following contributions:

- We propose two models that incorporate the social network into a Bayesian framework for matrix factorization. The first model, called Edge MRF, places the social network in the prior distribution. The second, called the Social Likelihood model, places the social network in the likelihood function. To the best of our knowledge, these are some of the first fully Bayesian matrix factorization models for recommendation in social networks.
- We perform experiments on a large scale, real world dataset obtained from the Flixster.com social network.
- We report state of the art predictive performance for the Social Likelihood model for cold start users.
- Based on our experimental results, we conclude that the Social Likelihood model is better for cold start users than placing the social network in the prior. The Social Likelihood model performs better in higher dimensions than the social prior alternatives, because the former relies on the same inner product structure that is used to predict ratings.

The rest of the paper is organized as follows. We present the probabilistic models and algorithms for inference in Section 3. Section 4 presents an evaluation of our models using the Flixster data set. Finally, Section 5 concludes the paper and discusses some potential future work.

2. SOCIAL LINKS

For any given social network $\mathcal{S}$ with links $(i, i') \in \mathcal{S}$ between users i and i' in a system, we aim to encode the similarity between the users' latent feature or *taste* vectors $\mathbf{u}_i$ and $\mathbf{u}_{i'} \in \mathbb{R}^K$ in a number of ways:

1. For each link, we define an "edge" energy

$$E(\mathbf{u}_i, \mathbf{u}_{i'}) = -\frac{\tau_{ii'}}{2}\|\mathbf{u}_i - \mathbf{u}_{i'}\|^2 \ , \tag{1}$$

which we incorporate into a Markov Random Field (MRF) prior distribution $p(\mathbf{U})$ over all user features. Furthermore, we may not know the connection strength $\tau_{ii'}$, and wish to infer that from user ratings; in other words, for users with dissimilar tastes we hope that $\tau_{ii'}$ is negligible.

2. The links can be treated as explicit observations: define $\ell_{ii'} = 1$ if $(i, i') \in \mathcal{S}$, and $\ell_{ii'} = -1$ otherwise. The system can treat $\ell_{ii'}$ as observations with

$$p(\ell_{ii'} \mid \mathbf{u}_i,\, \mathbf{u}_{i'}) = \Phi(\ell_{ii'}\, \mathbf{u}_i^T \mathbf{u}_{i'}) \ , \tag{2}$$

where $\Phi(z) = \int_{-\infty}^{x} \mathcal{N}(x; 0, 1)\, \mathrm{d}x$ is the cumulative Normal distribution. This likelihood is akin to a linear classification model, where an angle of less than 90° between $\mathbf{u}_i$ and $\mathbf{u}_{i'}$ gives likelihood greater than a half for the discrete value of $\ell_{ii'}$.

3. Let $\mathcal{S}(i) = \{i' : (i, i') \in \mathcal{S}\}$ be the set of neighbors for user i. Jamali and Ester [17] use an energy

$$E(\mathbf{u}_i, \mathbf{U}) = -\frac{\tau_J}{2}\left\|\mathbf{u}_i - \frac{1}{|\mathcal{S}(i)|}\sum_{i' \in \mathcal{S}(i)} \mathbf{u}_{i'}\right\|^2 \tag{3}$$

in the prior, which can also be folded into an MRF. The user feature is *a priori* expected to lie inside the convex hull of its neighbors' feature vectors.

3. PROBABILISTIC MODELS

We observe a user i's feedback on item j, which we denote by $r_{ij} \in \mathbb{R}$. Similar to the users, we let each item have a latent feature $\mathbf{v}_j \in \mathbb{R}^K$. Their combination produces the observed rating with noise,

$$p(r_{ij} \mid \mathbf{u}_i,\, \mathbf{v}_j) = \mathcal{N}(r_{ij}\, ;\, \mathbf{u}_i^T \mathbf{v}_j,\, \lambda^{-1}) \ . \tag{4}$$

We furthermore define $E(\mathbf{u}_i) = -\alpha_u \|\mathbf{u}_i\|^2/2$ and $E(\mathbf{v}_i) = -\alpha_v \|\mathbf{v}_j\|^2/2$, giving a Normal prior distribution on $\mathbf{V}$ as $p(\mathbf{V}) = \prod_i \mathcal{N}(\mathbf{v}_{ij}\, ;\, \mathbf{0},\, \alpha_v^{-1}\mathbf{I})$. In the "Social Likelihood" model the prior $p(\mathbf{U})$ would take the same form. However, in the MRF models we encode $\mathcal{S}$ in the prior with either

$$p(\mathbf{U}) \propto \exp\left[\sum_i E(\mathbf{u}_i) + \sum_{(i,i') \in \mathcal{S}} E(\mathbf{u}_i, \mathbf{u}_{i'})\right] \tag{5}$$

for the "Edge MRF" model or

$$p(\mathbf{U}) \propto \exp\left[\sum_i E(\mathbf{u}_i) + E(\mathbf{u}_i, \mathbf{U})\right] \tag{6}$$

for the "Average Neighbor" model. Both of these priors leave any user $\mathbf{u}_i | \mathbf{U}_{\setminus i}$ to be conditionally Gaussian (read $\setminus$ as *without*), and can easily be treated with Gibbs sampling.

We now observe a sparse matrix $\mathbf{R}$ with entries r_{ij}, and consider the models, and the conditional distributions of their random variables. The models under consideration are:

3.1 Baseline

Rating data in collaborative filtering systems generally exhibit large user and item effects that are independent of user-item interactions [19] expressed in the baseline model. For example, some users tend to give higher ratings than others, and some items tend to receive higher ratings than others. We model these effects with user and item biases, $\mathbf{b}_u$ and $\mathbf{b}_v$, respectively. With these biases, the conditional distribution for observed ratings becomes

$$p(r_{ij} \mid \mathbf{u}_i,\, \mathbf{v}_j,\, b_i,\, b_j) = \mathcal{N}(r_{ij}\, ;\, \mathbf{u}_i^T \mathbf{v}_j + b_i + b_j,\, \lambda^{-1}) \ . \tag{7}$$

where b_i is the bias for each user and b_j is the bias for each item. We place flexible hyperpriors on the precisions for user and item biases, denoted by α_{bu} and α_{bv}.

The baseline model depends on the settings λ, α_u, α_v, α_{bu}, and α_{bv}, and ignores the social network and any of the additions to the model that were described in Section 2. As the λ and α's are unknown, we place a flexible hyperprior – a conjugate Gamma distribution – on each, for example

$$p(\alpha_u) = \mathcal{G}(\alpha_u\, ;\, a_{u0},\, b_{u0}) = \frac{1}{\Gamma(a_{u0})} b_{u0}^{a_{u0}} \alpha^{a_{u0}-1} e^{-b_{u0}\alpha_u} \ .$$

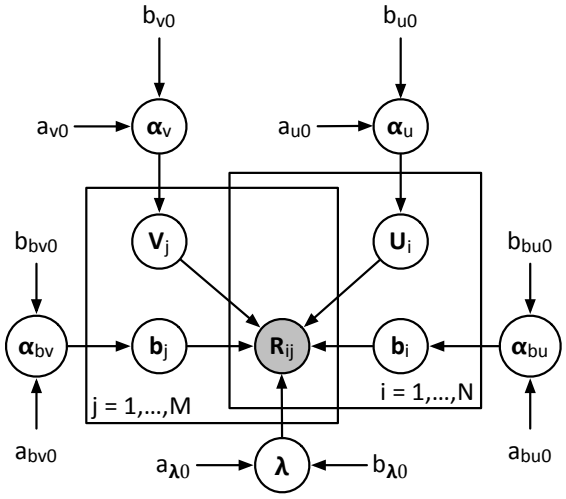

Figure 1: Graphical model for baseline model.

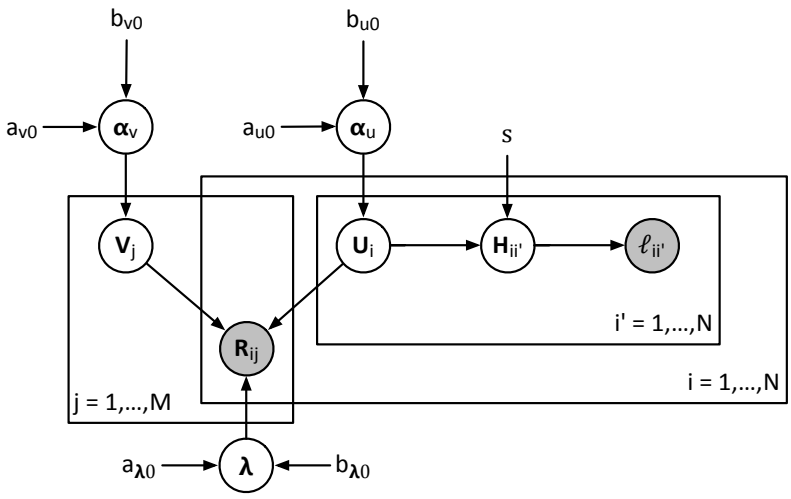

Figure 2: Graphical model for Social Likelihood model. Biases are omitted in this graphical model for clarity.

Figure 1 shows the graphical model for the baseline matrix factorization model.

Inference for all the models will be done through Gibbs sampling [14], which sequentially samples from the conditional distributions in a graphical model. The samples produced from the arising Markov chain are from the required posterior distribution if the chain is aperiodic and irreducible.

If we denote the entire set of *baseline* random variables with $\theta = \{\mathbf{U}, \mathbf{V}, \mathbf{b}_u, \mathbf{b}_v, \lambda, \alpha_u, \alpha_v, \alpha_{bu}, \alpha_{bv}\}$, then samples for $\mathbf{u}_i$ are drawn from

$$\begin{aligned}
\mathbf{u}_i|\mathbf{R}, \theta_{\setminus \mathbf{u}_i} &\sim \mathcal{N}(\mathbf{u}_i\,;\, \boldsymbol{\mu}_i,\, \boldsymbol{\Sigma}_i) \\
\boldsymbol{\mu}_i &= \boldsymbol{\Sigma}_i \left[\lambda \sum_{j \in \mathcal{R}(i)} (r_{ij} - (b_i + b_j))\mathbf{v}_j \right] \\
\boldsymbol{\Sigma}_i &= \left(\alpha_u \mathbf{I} + \lambda \sum_{j \in \mathcal{R}(i)} \mathbf{v}_j \mathbf{v}_j^T \right)^{-1}
\end{aligned} \tag{8}$$

We've defined $\mathcal{R}(i)$ as the set of items j rated by user i. A similarly symmetric conditional distribution holds for each $\mathbf{v}_j$.

The conditional distribution $b_i|\mathbf{R}, \theta^{\text{bias}}_{\setminus \mathbf{b}_i} \sim \mathcal{N}(b_i\,;\, \mu_i,\, \sigma_i^2)$ is

$$\begin{aligned}
\mu_i &= \sigma_i^2 \left[\lambda \sum_{j \in \mathcal{R}(i)} (r_{ij} - (\mathbf{u}_i^T \mathbf{v}_j + b_j)) \right] \\
\sigma_i^2 &= \left(\alpha_{bu} + \sum_{j \in \mathcal{R}(i)} \lambda \right)^{-1}
\end{aligned} \tag{9}$$

A similar conditional distribution holds for each b_j.

Due to the conveniently conjugate prior on α_u, its conditional distribution is also a Gamma density,

$$\begin{aligned}
\alpha_u|\theta_{\setminus \alpha_u} &\sim \mathcal{G}(\alpha_u\,;\, a_u,\, b_u) \\
a_u &= a_{u0} + \frac{|\mathcal{U}|K}{2} \\
b_u &= b_{u0} + \frac{1}{2} \sum_{i \in \mathcal{U}} \|\mathbf{u}_i\|^2
\end{aligned} \tag{10}$$

$\mathcal{U}$ is defined as the set of all users. A similarly symmetric conditional distribution holds for α_v.

The conditional distribution used for sampling α_{bu} is

$$\begin{aligned}
\alpha_{bu}|\theta_{\setminus \alpha_{bu}} &\sim \mathcal{G}(\alpha_{bu}\,;\, a_{bu},\, b_{bu}) \\
a_{bu} &= a_{bu0} + \frac{|\mathcal{U}|}{2} \\
b_{bu} &= b_{bu0} + \frac{1}{2} \sum_{i \in \mathcal{U}} b_i^2
\end{aligned} \tag{11}$$

A similarly symmetric conditional distribution holds for α_{bv}.

Finally, we draw samples for λ from

$$\begin{aligned}
\lambda|\theta_{\setminus \lambda} &\sim \mathcal{G}(\lambda\,;\, a_\lambda,\, b_\lambda) \\
a_\lambda &= a_{\lambda 0} + \frac{|\mathcal{R}|}{2} \\
b_\lambda &= b_{\lambda 0} + \frac{1}{2} \sum_{i,j \in \mathcal{R}} (r_{ij} - (\mathbf{u}_i^T \mathbf{v}_j + b_i + b_j))^2
\end{aligned} \tag{12}$$

We've defined $\mathcal{R}$ as the set of all ratings.

Algorithm 1 gives a pseudo-algorithm for sampling from $\theta|\mathbf{R}$.

The predicted rating $\hat{r}_{ij}$ for any user and item can be determined by averaging Equation (4) over the parameter posterior $p(\theta|\mathbf{R})$. If samples $\theta^{(t)}$ are simulated from the posterior distribution, this average is approximated with the Markov chain Monte Carlo (MCMC) estimate

$$\hat{r}_{ij} = \frac{1}{T} \sum_t ({\mathbf{u}_i^{(t)}}^T \mathbf{v}_i^{(t)} + \mathbf{b}_i^{(t)} + \mathbf{b}_j^{(t)}) \ .$$

3.2 Edge MRF

The Edge MRF model uses the prior in (5), which additionally depends on the setting of $\tau_{ii'}$ for all $(i, i') \in \mathcal{S}$. If the $\tau_{ii'}$ parameters are flexible, we hope to infer that the *similarity connection* between two users with vastly different ratings should be negligible, while correlations in very similar users should be reflected in a higher $\tau_{ii'}$ connection between them.

We extend the set of random variables to $\theta^{\text{edge}} = \{\theta, \boldsymbol{\tau}\}$. Due to $\tau_{ii'}$ now appearing in $\mathbf{u}_i$'s Markov blanket in Figure 3,

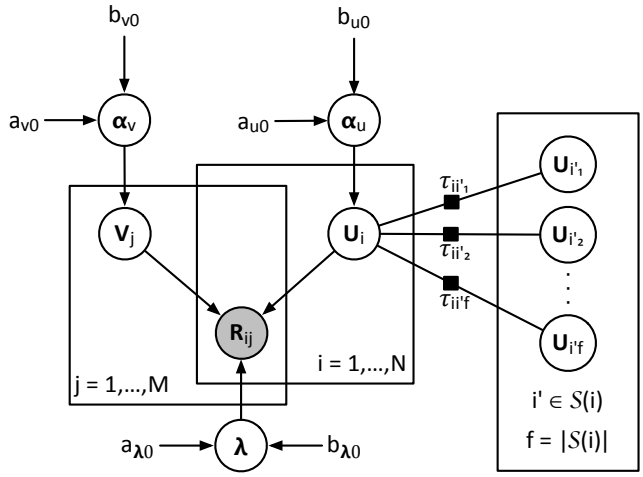

Figure 3: Graphical model for Edge MRF model. Biases are omitted in this graphical model for clarity.

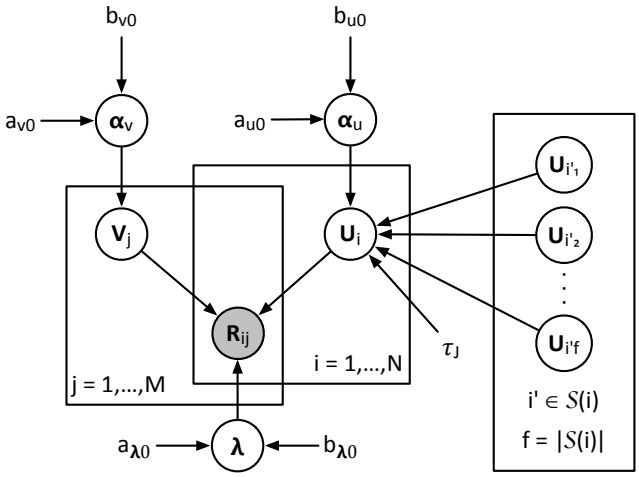

Figure 4: Graphical model for Average Neighbor model. Biases are omitted in this graphical model for clarity.

the conditional distribution for $\mathbf{u}_i$ changes to the Gaussian $\mathbf{u}_i|\mathbf{R}, \theta^{\text{edge}}_{\setminus \mathbf{u}_i} \sim \mathcal{N}(\mathbf{u}_i\,;\,\boldsymbol{\mu}_i,\,\boldsymbol{\Sigma}_i)$ with

$$\boldsymbol{\mu}_i = \boldsymbol{\Sigma}_i \left[\sum_{i' \in \mathcal{S}(i)} \tau_{ii'} \mathbf{u}_{i'} + \lambda \sum_{j \in \mathcal{R}(i)} r_{ij} \mathbf{v}_j \right]$$

$$\boldsymbol{\Sigma}_i = \left(\alpha_u \mathbf{I} + \lambda \sum_{j \in \mathcal{R}(i)} \mathbf{v}_j \mathbf{v}_j^T + \sum_{i' \in \mathcal{S}(i)} \tau_{ii'} \mathbf{I} \right)^{-1} . \quad (13)$$

There is an interplay in $\boldsymbol{\mu}_i$ above, where $\mathbf{u}_i$ is a combination of his neighbors $\mathbf{u}_{i'}$, and items rated $\mathbf{v}_j$.

By placing a flexible Gamma prior independently on each $\tau_{ii'}$, we can infer each individually with

$$\begin{aligned} \tau_{ii'}|\theta^{\text{edge}}_{\setminus \tau_{ii'}} &\sim \mathcal{G}(\tau_{ii'}\,;\,a_\tau,\,b_\tau) \\ a_\tau &= a_{\tau 0} + \frac{1}{2} \\ b_\tau &= b_{\tau 0} + \frac{1}{2}\|\mathbf{u}_i - \mathbf{u}_{i'}\|^2 \end{aligned} \quad (14)$$

3.3 Social Likelihood

Instead of embedding $\mathcal{S}$ in the prior distribution, we can treat it as observations that need to be modeled together with $\mathbf{R}$. To adjust for the fact that there might be an imbalance between the two observations (for example, $|\mathcal{S}|$ might be much larger than the number of observed ratings) we introduce an additional knob $s > 0$ in the likelihood. When the graphical model only needs to explain observations $\ell_{ii'} = 1$, the inclusion of $\mathcal{S}$ shouldn't outweigh any evidence provided by the user ratings. Hence

$$p(\ell_{ii'} \,|\, \mathbf{u}_i,\, \mathbf{u}_{i'}) = \Phi(\ell_{ii'}\, s\, \mathbf{u}_i^T \mathbf{u}_{i'}) \,. \quad (15)$$

The effect of the likelihood is that $\mathbf{u}_i$ and $\mathbf{u}_{i'}$ should lie on the same side of a hyperplane perpendicular to either, like a linear classification model.

We extend the set of random variables to $\theta^{\text{sl}} = \{\theta, \mathbf{H}\}$. $\mathbf{H}$ is a set of latent variables that make sampling from the likelihood possible, and contains an $h_{ii'} = s\mathbf{u}_i^T \mathbf{u}_{ii'} + \epsilon$ with $\epsilon \sim \mathcal{N}(0, 1)$. We give its updates in the Appendix.

Again, the conditional distribution of $\mathbf{u}_i$ will adapt according to the additions in the graphical model, shown in Figure 2. $\mathbf{u}_i|\mathbf{R}, \mathcal{S}, \theta^{\text{sl}}_{\setminus \mathbf{u}_i}$ is

$$\boldsymbol{\mu}_i = \boldsymbol{\Sigma}_i \left[s \sum_{i' \in \mathcal{S}(i)} h_{ii'} \mathbf{u}_{i'} + \lambda \sum_{j \in \mathcal{R}(i)} r_{ij} \mathbf{v}_j \right]$$

$$\boldsymbol{\Sigma}_i = \left(\alpha_u \mathbf{I} + \lambda \sum_{j \in \mathcal{R}(i)} \mathbf{v}_j \mathbf{v}_j^T + s \sum_{i' \in \mathcal{S}(i)} \mathbf{u}_{i'} \mathbf{u}_{i'}^T \right)^{-1} \quad (16)$$

The Social Likelihood model differs from both the Edge MRF and Average Neighbor models through real-valued latent variables $\mathbf{h}_{ii'}$. If we compare (16) to (13) and (17), we notice that $\mathbf{u}_i$ is no longer required to be a positive combination of its neighbors $\mathbf{u}_{i'}$. Indeed, if $\mathbf{u}_i$ and $\mathbf{u}_{i'}$ continually give opposite ratings to items, $h_{ii'}$ would be *negative*. When we compare $\boldsymbol{\mu}_i$ in Equation 16 with that of the Edge MRF model (Equation 13), we see that the Social Likelihood model places neighboring users that are similar to each other in the trait (latent feature) space on the same preference cone/hyperplane in this space, since the $h_{ii'}$ term in $\boldsymbol{\mu}_i$ is computed using the inner product between a user's feature vector and his neighbor's feature vector. In contrast, the Edge MRF model places neighboring users close to each other based on the Euclidean distance between their feature vectors, as we see from Equation 1. As we will observe from the experimental results in Section 4.3, this distinction between these models has an important impact on predictive performance.

3.4 Average Neighbor

An alternative to specifying a "spring" between the $\mathbf{u}_i$'s is to constrain each user's latent trait to be an average of those of his friends [17]. The maximum likelihood framework by Jamali and Ester [17] easily slots into the Gibbs sampler in Algorithm 1 by using the energy function (3) in the user prior (6). We add a fixed tunable scale parameter, $\tau_J > 0$, to the prior as shown in Figure 4, and extend the parameters

Algorithm 1 Gibbs sampling

initialize $\mathbf{U}$, $\mathbf{V}$, $\mathbf{b}_u$, $\mathbf{b}_v$, λ, α_u, α_v, α_{bu}, α_{bv}
if edge mrf **then**
  initialize $\tau_{ii'}$ for all $(i, i') \in \mathcal{S}$
end if
// gibbs sampling
repeat
  for items $j = 1, \ldots, J$ in random order **do**
    sample $\mathbf{v}_j$, similar to (8)
    sample $\mathbf{b}_j$, similar to (9)
  end for
  for users $i = 1, \ldots, I$ in random order **do**
    if baseline **then**
      sample $\mathbf{u}_i$ according to (8)
      sample $\mathbf{b}_i$ according to (9)
    else if edge mrf **then**
      sample $\tau_{ii'}$ for each $i' \in \mathcal{S}(i)$ according to (14)
      sample $\mathbf{u}_i$ according to (13)
    else if social likelihood **then**
      sample $h_{ii'}$ for each $i' \in \mathcal{S}(i)$ according to the Appendix
      sample $\mathbf{u}_i$ according to (16)
    else
      // average neighbor
      sample $\mathbf{u}_i$ according to (17)
    end if
  end for
  sample α_u according to (10)
  sample α_v similar to (10)
  sample α_{bu} according to (11)
  sample α_{bv} similar to (11)
  sample λ according to (12)
until sufficient samples have been taken

to $\theta^{\text{an}} = \{\theta, \tau_J\}$. The conditional density $\mathbf{u}_i|\mathbf{R}, \mathcal{S}, \theta^{\text{an}}_{\setminus \mathbf{u}_i}$ is

$$\boldsymbol{\mu}_i = \boldsymbol{\Sigma}_i \left[\tau_J \sum_{i' \in \mathcal{S}(i)} \frac{1}{|\mathcal{S}(i)|} \mathbf{u}_{i'} + \lambda \sum_{j \in \mathcal{R}(i)} r_{ij} \mathbf{v}_j \right]$$
$$\boldsymbol{\Sigma}_i = \left(\alpha_u \mathbf{I} + \lambda \sum_{j \in \mathcal{R}(i)} \mathbf{v}_j \mathbf{v}_j^T + \tau_J \mathbf{I} \right)^{-1} \quad (17)$$

We do not sample for τ_J because there is no closed-form expression for the conditional density on τ_J. Therefore, because of the difficulty of sampling from this conditional density, we treat τ_J as a tunable fixed parameter.

The Average Neighbor model differs from the Edge MRF model in that each user's feature vector is constrained to be the average of the feature vector of his neighbors. This difference is apparent when comparing the first term of $\boldsymbol{\mu}_i$ in Equation 17 and Equation 13. By constraining the user's feature vector to the average of his neighbors, we allow for less flexibility in learning the user's feature vector as compared to the flexible, independent $\tau_{ii'}$ for each of the user's social links. In the Average Neighbor model, each of a user's neighbors contributes equally to the user's feature vector, while in the Edge MRF model, the extent of each neighbor's contribution varies based on the similarity between the user and neighbor as expressed by $\tau_{ii'}$.

4. EVALUATION

We evaluated the four models described in Section 3 by evaluating their predictive performance on a publicly available data set obtained from the Flixster.com social networking Web site [5]. In this section we describe the Flixster data set, our experimental setup, and the results of our performance experiments.

Metric	Flixster
Users	1M
Social Links	5.8M
Ratings	8.2M
Items	49K
Users with Rating	130K
Users with Friend	790K

Table 1: General metrics for the Flixster data set

4.1 Flixster Data Set

Flixster is an online social network (OSN) that allows users to rate movies, share movie ratings, discover new movies, and add other users as friends. Each movie rating in Flixster is a discrete value in the range [0.5, 5] with a step size of 0.5, so there are ten possible rating values (0.5, 1.0, 1.5, ...). To our knowledge, the Flixster data set we use is the largest publicly available OSN data set that contains numeric ratings for items. We show some general metrics for the Flixster dataset in table 1.

4.2 Experimental Setup

The metric we use to evaluate predictive performance is root mean square error (RMSE), which is defined as

$$RMSE = \sqrt{\frac{\sum_{(i,j)} (r_{i,j} - r'_{i,j})^2}{n}} \quad (18)$$

where $r_{i,j}$ is the actual rating for user i and item j from the test set, $r'_{i,j}$ is the predicted rating, and n is the number of ratings in the test set. We randomly select 80% of the Flixster data as the training set and the remaining as the test set.

In all of our experiments, we place flexible priors on the α's and λ in our models by setting $a_{u0} = a_{v0} = \sqrt{K}$, $b_{u0} = b_{v0} = b_{bu0} = b_{bv0} = 1$, $a_{bu0} = a_{bv0} = 2$, and $a_{\lambda 0} = b_{\lambda 0} = 1.5$. For the Edge MRF model, we place a flexible prior on each $\tau_{ii'}$ by setting $a_{\tau 0} = b_{\tau 0} = 0.15$. We set $s = 1$ for the Social Likelihood model and $\tau_J = 1$ for the Average Neighbor model in all experiments, except where s and τ_J are adjusted between a range of 0.001 and 1000 as stated below.

We run the Gibbs samplers for all of our experiments with a burn-in of 50 update samples of all parameters and 300 samples after burn-in. During the post burn-in period of 300 samples, we collect samples for all parameters and compute updated predictions based on each sample. Figure 5 shows RMSE as the number of samples increases for each model. The Gibbs samplers converge quickly, and after obtaining 200-250 samples, the predictive performance does not significantly improve.

4.3 Experimental Results

Table 2 shows the RMSE values for all of our models for different settings of the latent factor dimensionality parameter K. We see that predictive performance generally improves (i.e., RMSE decreases) as K is increased, as expected. Notice that predictive performance is relatively close amongst all models for $K = 5$, to within 0.29%, while the performance delta increases to 0.76% for $K = 20$. This may

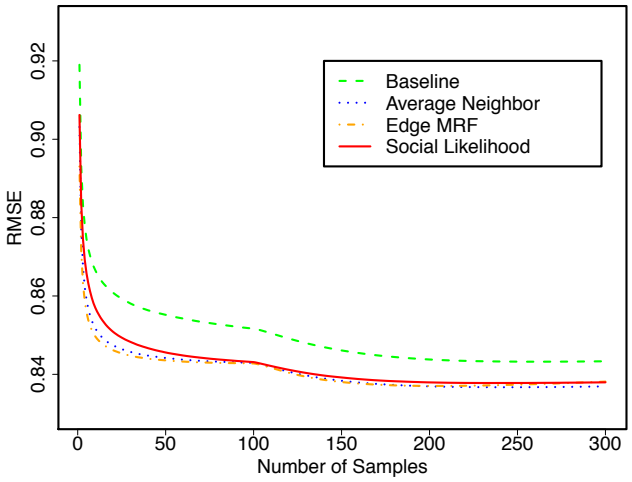

Figure 5: RMSE for models as a function of the number of samples included in the estimate, after burn-in.

Model	$K = 5$	$K = 10$	$K = 20$
Baseline	0.8590	0.8468	0.8433
Edge MRF	0.8593	0.8458	0.8381
Average Neighbor	0.8581	0.8423	0.8369
Social Likelihood	0.8568	0.8442	0.8380

Table 2: RMSE for models with different settings of dimensionality K

indicate that the social-based models are able to more effectively exploit social network signals as the number of model parameters increases, which is not possible for the baseline model with no consideration of the social network.

Next, we examine the predictive performance of our models for cold start users. We define cold start users as users who have rated five movies or less in the training set. For the Flixster data set approximately 40% of users with ratings are cold start users, so predictive performance on cold start users is quite important.

Table 3 shows the RMSE values for our models for cold start users. Notice that for the baseline model, predictive performance worsens as K is increased. In contrast, the other models provide approximately the same or improved predictive performance as model complexity grows. Based on these results, we see that for cold start users, the Social Likelihood model provides the best predictive performance of the models that we considered, and that all of the social models outperform the baseline model. Furthermore, compared to the baseline model, we conclude that the Edge MRF and Average Neighbor models place a more effective prior distribution on the model parameters by considering the social network. However, these models are outperformed by the Social Likelihood model, which is able to effectively model the social network as observations using the inner product similarity between neighbors. Figures 7, 8, and 9 reveal that the performance differences between models tend to be minimized as the number of observed ratings per user increases.

Figure 6 shows that the Social Likelihood model outperforms the other models for users with few ratings (10 or less ratings). As the number of ratings increases, the predictive performance of all models converges. Based on the results presented in Table 3 and Figure 6, we conclude that for cold start users, the Social Likelihood model is able to leverage the social network more effectively than the other models we considered. For users with more ratings, the social network appears to have little to no impact on predictive performance.

Model	$K = 5$	$K = 10$	$K = 20$
Baseline	1.1205	1.14407	1.2180
Edge MRF	1.1424	1.0970	1.0984
Average Neighbor	1.0814	1.0721	1.0662
Social Likelihood	1.0569	1.0583	1.0563

Table 3: RMSE for cold start users for models with different settings of dimensionality K

Recall that the s parameter controls the influence of the social network in the Social Likelihood model. Larger values of s cause the social network to have more influence on the learned latent feature vectors for users, while smaller values of s cause the social network to have less impact. Figure 10 compares the predictive performance of the Social Likelihood model for different values of s for users with few (0-5), more (40-80), and many ratings (320-640). These results show that s has little impact on predictive performance for users with more ratings. However, for cold start users with 0-5 ratings, s has a significant impact on predictive performance. For these users, the optimal value of s appears to be approximately 1. These findings regarding the impact of s on cold start users vs. users with more ratings are in agreement with our other results. Therefore, we conclude that the social network has a significant impact on predictive performance only for cold start users.

In the Average Neighbor model, the τ_J parameter controls the influence of the social network. Figure 11 compares the predictive performance of the Average Neighbor model for different values of τ_J for users with few (0-5), more (40-80), and many ratings (320-640). These results show that τ_J has little impact on predictive performance for users with more ratings. However, for cold start users with 0-5 ratings, τ_J has a significant impact on predictive performance. For these users, the optimal value of τ_J appears to be approximately 1. These results are similar to the findings for the impact of the s parameter in the Social Likelihood model, and provide further evidence that the social network has a significant impact on predictive performance only for cold start users.

In Figure 12, we examine how predictive performance of the Social Likelihood model changes with the number of observed friends (neighbors) per user, for users with few (0-20), more (60-160), and many (200 or more) friends. We see that predictive performance for cold start users is best when these users have many friends. When the number of observed ratings per user exceeds 320 ratings, we see that the predictive performance is worst for users with many friends. Therefore, we conclude that for a user with many ratings and many friends, when we consider this user's observed ratings and social network, the observed ratings data can be a better indicator of the user's preferences.

5. CONCLUSIONS

In this work we have proposed and investigated two novel models for including a social network in a Bayesian framework for recommendation using matrix factorization. The first model, which we call the Edge MRF model, places the

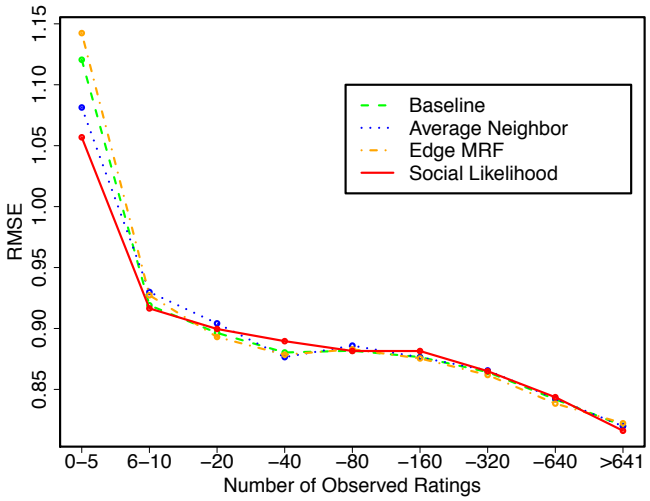

Figure 6: Performance of models, where users are grouped by the number of observed ratings in the training data. These results were obtained using $K = 5$ models.

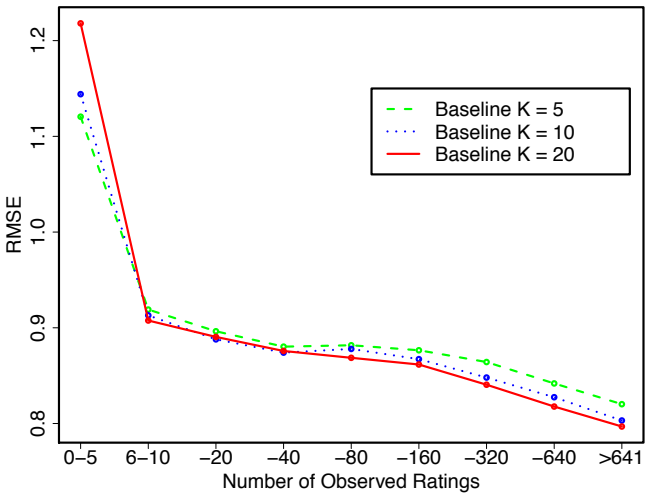

Figure 7: Performance of baseline models, where users are grouped by the number of observed ratings in the training data. These results were obtained using $K = 5$, 10, and 20 models.

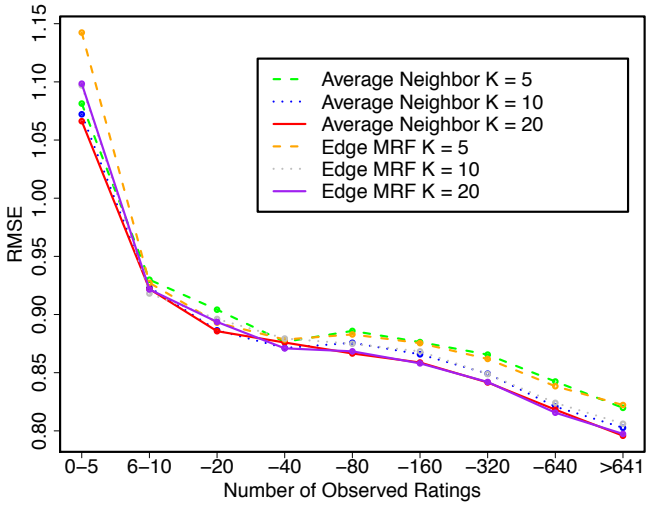

Figure 8: Performance of Edge MRF and Average Neighbor models, where users are grouped by the number of observed ratings in the training data. These results were obtained using $K = 5$, 10, and 20 models.

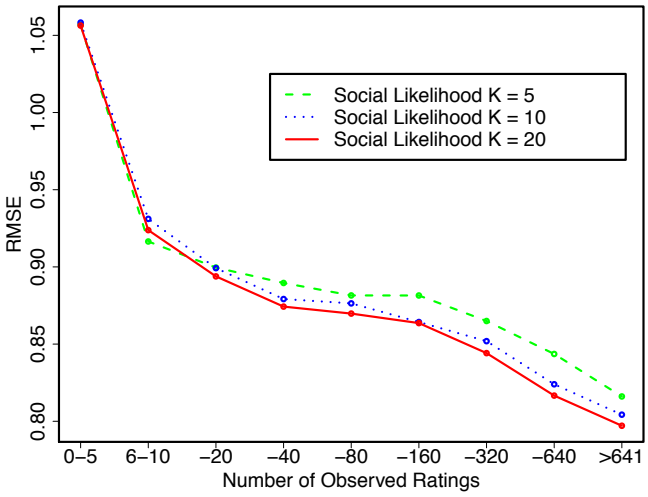

Figure 9: Performance of Social Likelihood models, where users are grouped by the number of observed ratings in the training data. These results were obtained using $K = 5$, 10, and 20 models.

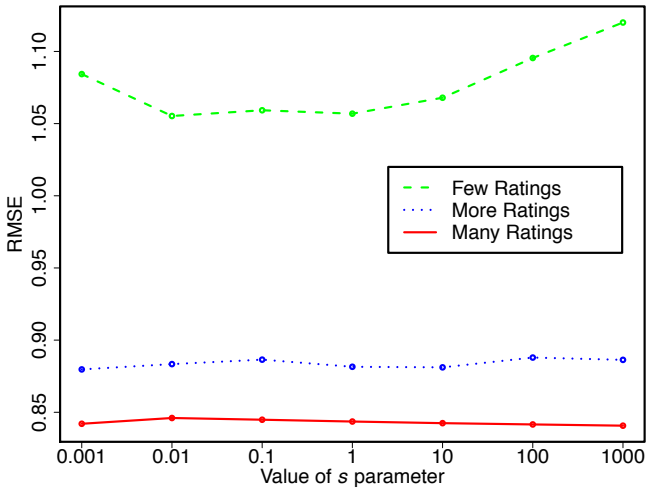

Figure 10: Impact of the value of s on the predictive performance for users with few (0-5), more (40-80), and many (320-640) ratings. Results were obtaining using the Social Likelihood model with $K = 5$.

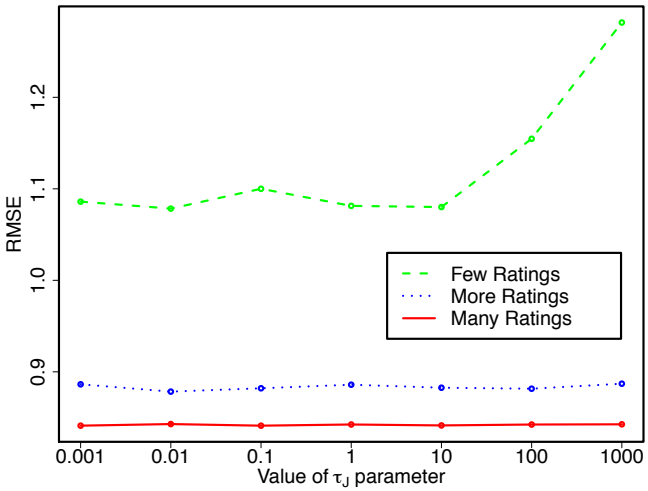

Figure 11: Impact of the value of τ_J on the predictive performance for users with few (0-5), more (40-80), and many (320-640) ratings. Results were obtaining using the Average Neighbor model with $K = 5$.

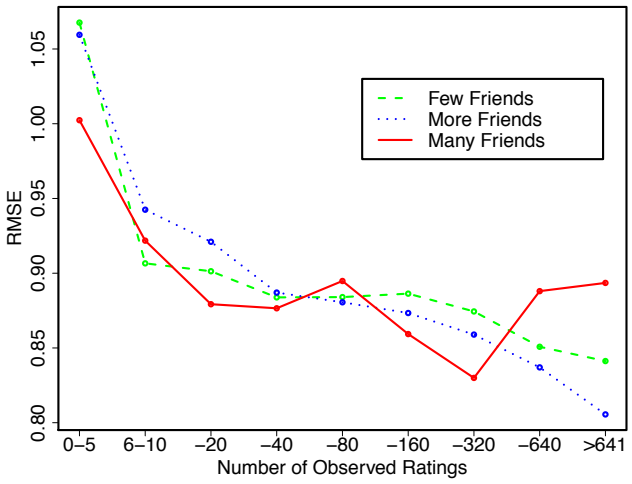

Figure 12: Impact of the number of observed friends per user on the predictive performance for users with few (0-20), more (60-160), and many (200 or more) friends. In addition to the number of friends, users are grouped by the number of observed ratings in the training data. Results were obtaining using the Social Likelihood model with $K = 5$.

social network in the prior distribution over user features as a Markov Random Field that describes user similarity. The second model, called the Social Likelihood model, treats social links as observations and places the social network in the likelihood function. We evaluate both models using a large scale dataset collected from the Flixster online social network. Experimental results indicate that while both models perform well, the Social Likelihood model outperforms existing methods for recommendation in social networks when considering cold start users who have rated few items.

6. REFERENCES

[1] Amazon. `http://www.amazon.com`.
[2] Facebook. `http://www.facebook.com`.
[3] The facebook blog - our first 100 million. `http://blog.facebook.com/blog.php?post=28111272130`.
[4] Facebook now has 800 million users. `http://mashable.com/2011/09/22/facebook-800-million-users/`.
[5] Flixster data set. `http://www.cs.sfu.ca/~sja25/personal/datasets/`.
[6] Google+. `http://plus.google.com`.
[7] Google news. `http://news.google.com`.
[8] Linkedin. `http://www.linkedin.com`.
[9] The music genome project. `http://www.pandora.com/mgp.shtml`.
[10] Netflix. `http://www.netflix.com`.
[11] Pandora. `http://www.pandora.com`.
[12] J. Breese, D. Heckerman, C. Kadie, et al. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.
[13] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.
[14] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):721–741, 1984.
[15] J. Golbeck. *Computing and Applying Trust in Web-based Social Networks*. PhD thesis, University of Maryland College Park, 2005.
[16] M. Jamali and M. Ester. Trustwalker: a random walk model for combining trust-based and item-based recommendation.

In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 397–406, New York, NY, USA, 2009. ACM.

[17] M. Jamali and M. Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys 2010, pages 135–142, New York, NY, USA, 2010. ACM.

[18] R. Jin, J. Chai, and L. Si. An automatic weighting scheme for collaborative filtering. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 337–344. ACM, 2004.

[19] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[20] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[21] H. Ma, I. King, and M. R. Lyu. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR 2009, pages 203–210, New York, NY, USA, 2009. ACM.

[22] H. Ma, H. Yang, M. R. Lyu, and I. King. Sorec: social recommendation using probabilistic matrix factorization. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM 2008, pages 931–940, New York, NY, USA, 2008. ACM.

[23] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, WSDM 2011, pages 287–296, New York, NY, USA, 2011. ACM.

[24] P. Massa and P. Avesani. Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems*, RecSys '07, pages 17–24, New York, NY, USA, 2007. ACM.

[25] U. Paquet, B. Thomson, and O. Winther. A hierarchical model for ordinal matrix factorization. *Statistics and Computing*, pages 1–13, 2011.

[26] J. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*, pages 713–719. ACM, 2005.

[27] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th international conference on Machine learning*, pages 880–887. ACM, 2008.

[28] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. *Advances in neural information processing systems*, 20:1257–1264, 2008.

[29] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[30] N. Srebro and T. Jaakkola. Weighted low-rank approximations. In *Proceedings of 20th International Conference on Machine Learning*, volume 20, pages 720–727, 2003.

[31] L. Yu, R. Pan, and Z. Li. Adaptive social similarities for recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*, RecSys 2011, pages 257–260, New York, NY, USA, 2011. ACM.

[32] C. Ziegler. *Towards Decentralized Recommender Systems*. PhD thesis, University of Freiburg, 2005.

APPENDIX

Let $\mu = s\mathbf{u}_i^T\mathbf{u}_{i'}$ denote the inner product in (15), where

$$\Phi(\ell_{ii'}\,\mu) = \int \Theta(\ell_{ii'}\,h_{ii'})\,\mathcal{N}(h_{ii'};\mu,1)\,\mathrm{d}h_{ii'}$$

arises from marginalizing out latent variable $h_{ii'}$ from the joint density

$$p(\ell_{ii'}|h_{ii'})p(h_{ii'}|\mu) = \Theta(\ell_{ii'}\,h_{ii'})\,\mathcal{N}(h_{ii'};\mu,1)\ .$$

The step function $\Theta(x)$ is one when its argument is nonnegative, and zero otherwise. We wish to sample from the density $p(h_{ii'}|\ell_{ii'},\mu)$ to use in (16). We do so by first defining $\Phi_{\max} = 1$ and $\Phi_{\min} = \Phi(-\mu)$ if $\ell_{ii'} = 1$; alternatively, we set $\Phi_{\max} = \Phi(-\mu)$ and $\Phi_{\min} = 0$ if $\ell_{ii'} = -1$. We then sample $u \sim \mathcal{U}(\Phi_{\max} - \Phi_{\min})$, where $\mathcal{U}(\cdot)$ gives a uniform random number between zero and its argument.

A sample for $h_{ii'}$ is obtained through the transformation

$$h_{ii'} = \mu + \Phi^{-1}(\Phi_{\min} + u)\ .$$

Care should be taken with the numeric stability of Φ^{-1} when its arguments are asymptotically close to zero or one; see [25] for further details.