Sparse Linear Methods with Side Information for Top-N Recommendations

Xia Ning
Computer Science & Engineering
University of Minnesota, Twin Cities
4-192 EE/CS Building, 200 Union Street SE
Minneapolis, MN 55455
xning@cs.umn.edu

George Karypis
Computer Science & Engineering
University of Minnesota, Twin Cities
4-192 EE/CS Building, 200 Union Street SE
Minneapolis, MN 55455
karypis@cs.umn.edu

ABSTRACT

The increasing amount of side information associated with the items in E-commerce applications has provided a very rich source of information that, once properly exploited and incorporated, can significantly improve the performance of the conventional recommender systems. This paper focuses on developing effective algorithms that utilize item side information for *top-N* recommender systems. A set of sparse linear methods with side information (`SSLIM`) is proposed, which involve a regularized optimization process to learn a sparse aggregation coefficient matrix based on both user-item purchase profiles and item side information. This aggregation coefficient matrix is used within an item-based recommendation framework to generate recommendations for the users. Our experimental results demonstrate that `SSLIM` outperforms other methods in effectively utilizing side information and achieving performance improvement.

Categories and Subject Descriptors

H.4.m [**Information Systems**]: Miscellaneous; J.7 [**Computer Applications**]: Computers in other systems—*Consumer products*

Keywords

Recommender system, Sparse Linear Methods, Side information

1. INTRODUCTION

Top-N recommender systems have been widely used in E-commerce applications to recommend ranked lists of items so as to help the users in identifying the items that best fit their personal tastes. Over the years, various algorithms for *top-N* recommendation have been developed [12]. The conventional *top-N* recommendation algorithms primarily focus on utilizing user-item purchase profiles to generate recommendations. Such algorithms can be categorized into two classes: collaborative filtering methods and model-based methods. Collaborative filtering methods typically build neighborhood for each user/item by considering the similarities of the purchase patterns among users/items from their profiles, and then recommend new items from the neighborhood. Model-based methods learn to explain the user-item purchase patterns using a specific model. For instance, the most popular matrix factorization (`MF`) methods present users and items in a common latent space such that the user-item purchase patterns can be explained by the user-item similarities in the space. Recently, a sparse linear method (`SLIM`) [10] has been developed that leverages the advantages of the above two classes of methods and achieves both better prediction accuracy and run-time performance than the state-of-the-art methods.

With the increased availability of additional information associated with the items (e.g., product reviews, movie plots, etc), referred to as *side information*, there is a greater interest in taking advantage of such information to improve the quality of conventional *top-N* recommender systems. As a result, a number of approaches have been developed from Machine Learning (ML) and Information Retrieval (IR) communities for incorporating side information. Such approaches include hybrid methods [5], matrix/tensor factorization [14, 8], and other regression methods [1].

In this paper, we propose a set of `S`parse `L`inear `M`ethods that utilize the item `S`ide information (`SSLIM`) for *top-N* recommendation. These methods include collective `SLIM` (`cSLIM`), relaxed collective `SLIM` (`rcSLIM`), side information induced `SLIM` (`fSLIM`) and side information induced double `SLIM` (`f2SLIM`). The key idea behind these methods is to learn linear models that are constrained and/or informed by the relations between the item side information and the user-item purchase profiles so as to achieve better recommendation performance. We conduct a comprehensive set of experiments on various datasets from different real applications. The results show that `SSLIM` produces better recommendations than the state-of-the-art methods.

The rest of this paper is organized as follows. In Section 2, a brief review on related work is presented. In Section 3, the definitions and notations are provided. In Section 4 and Section 5, the methods are described. In Section 6, the materials used for experiments are presented. In Section 7, the results are presented. Finally in Section 8 are the discussions and conclusions.

2. RELATED WORK

RecSys 2012 Dublin, Ireland

Various methods have been developed to incorporate side information in recommender systems. Most of these methods have been developed in the context of the rating prediction problem, whereas the *top-N* recommendation problem has received less attention. In the rest of this section we review some of the best performing schemes for both the rating prediction and *top-N* recommendation problems.

The first category of these methods is based on latent factor models. In [14], Singh *et al* proposed the collective matrix factorization method for both rating prediction and *top-N* recommendation, which collectively factorizes user-item purchase profile matrix and item-feature content matrix into a common latent space such that the two types of information are leveraged via common the item factors. Agarwal *et al* [1] proposed regression-based latent factor models for rating prediction, which use features for factor estimation. In their method, the user and item latent factors are estimated through independent regression on user and item features, and the recommendation is calculated from a multiplicative function on user and item factors. Yang *et al* [16] developed a joint friendship and interest propagation model for *top-N* recommendation, in which the user-item interest network and the user-user friendship network (side information on users) are jointly modeled through latent user and item factors. User factors are shared by the interest network approximation component and the friendship network approximation component so as to enable information propagation. They demonstrated the their model is a generalization of Singh *et al* [14], Koren [9] and Agarwal *et al* [1].

Methods using tensor factorization (`TF`) have also gained popularity. Karatzoglou *et al* [8] considered the user-item-feature relation as a tensor, and they proposed to use regularized `TF` to model the relations between the three sets of entities for rating prediction. `TF` can be considered as a generalization of `MF`, in which the relations among all the modes (i.e., users, items and features) are jointly learned. Rendle *et al* [11] also treated user-item-feature as a tensor, and they factorized all pairwise interactions in the tensor (i.e, items vs users, items vs context features, users vs context features) rather than the entire tensor for rating prediction.

Another category of algorithms that utilize side information is based on networks. Gunawardana *et al* [6] proposed unified Boltzmann machines for *top-N* prediction, in which user-item profile information and side information are treated as homogeneous features, and interaction weights between such features and user actions are learned in a coherent manner so as to reflect how well such features can predict user actions. Campos *et al* [3] combined content-based and collaborative-filtering based recommendations with Bayesian networks, which are composed of item nodes, user nodes and item feature nodes. During predictions, content information is propagated from purchased items to non-purchased items via feature nodes, and purchase information is propagated from other users to the user of concern via item nodes. Then the two parts are combined to make recommendations.

3. DEFINITIONS AND NOTATIONS

In this paper, the symbols u, t, and $\mathbf{f}$ ($|\mathbf{f}| = l$) will be used to denote the users, items and item side information vectors, respectively. Individual users and items will be denoted using different subscripts (i.e., u_i, t_j). The side information vector for item t_j will be denoted by $\mathbf{f}_j$. The size of user set and item set are denoted by m and n, respectively.

The user-item purchase profile is represented by a binary $m \times n$ matrix M, in which the (i, j) entry (denoted by $m_{i,j}$) is 1 if user u_i has ever purchased item t_j, otherwise it is marked as 0. The i-th row of M, denoted by $\mathbf{m}_i^\mathsf{T}$, represents the purchase profile of user u_i on all items. The j-th column of M, denoted by $\mathbf{m}_j$, represents the purchase profile of item t_j from all users. The side information on all items is represented by an $l \times n$ matrix F. The j-th column of F represents the side information vector of item t_j (i.e., $\mathbf{f}_j$).

All vectors (e.g., $\mathbf{m}_i^\mathsf{T}$ and $\mathbf{f}_j$) are represented by bold lower-case letters and all matrices (e.g., M and F) are represented by upper-case letters. Row vectors are represented by having the transpose supscript$^\mathsf{T}$, otherwise by default they are column vectors. Approximation relation is denoted using $\sim$ and approximation value is denoted by heading a $\sim$ head. The matrix/vector notations are used instead of user/item/side information if no ambiguity is raised.

4. SLIM: SPARSE LINEAR METHODS

In this paper, we focus on incorporating item side information within the Sparse LInear Method (`SLIM`) that we proposed previously [10]. In `SLIM`, the recommendation score on an urn-purchased item t_j of a user u_i (denoted by $\tilde{m}_{ij}$) is calculated as a *sparse* aggregation of the items that have been purchased by u_i, that is,

$$\tilde{m}_{ij} = \mathbf{m}_i^\mathsf{T} \mathbf{s}_j, \tag{1}$$

where $m_{ij} = 0$ and $\mathbf{s}_j$ is a size-n sparse vector of aggregation coefficients. Thus, the model can be presented as

$$M \sim MS, \tag{2}$$

where S is an $n \times n$ sparse matrix of aggregation coefficients, whose j-th column is $\mathbf{s}_j$ as in Equation 1, and each row $\tilde{\mathbf{m}}_i^\mathsf{T}$ of $\tilde{M} = MS$ represents the recommendation scores on all items for user u_i. The *top-N* recommendations for u_i are obtained by sorting u_i's non-purchased items based on their scores in $\tilde{\mathbf{m}}_i^\mathsf{T}$ in decreasing order and recommending the *top-N* items.

The `SLIM` method views the purchase activity of user u_i on item t_j in M (i.e., m_{ij}) as the ground-truth item recommendation score. It learns the $n \times n$ sparse matrix S in Equation 2 as the minimizer for the following regularized optimization problem:

$$\begin{aligned} \underset{S}{\text{minimize}} \quad & \frac{1}{2}\|M - MS\|_F^2 + \frac{\beta}{2}\|S\|_F^2 + \lambda\|S\|_1 \\ \text{subject to} \quad & S \geq 0 \\ & \text{diag}(S) = 0, \end{aligned} \tag{3}$$

where $\|S\|_1 = \sum_{i=1}^{n}\sum_{j=1}^{n}|s_{ij}|$ is the entry-wise ℓ_1-norm of S, and $\|\cdot\|_F$ is the matrix Frobenius norm. In Equation 3, MS is the estimated matrix of recommendation scores (i.e., $\tilde{M}$) by the sparse linear method as in Equation 2. The non-negativity constraint is applied on S such that the learned S corresponds to positive aggregations over items. The constraint $\text{diag}(S) = 0$ is also applied so as to avoid trivial solutions (i.e., the optimal S is an identity matrix such that an item always recommends itself). In addition, the constraint $\text{diag}(S) = 0$ ensures that m_{ij} is not used to compute $\tilde{m}_{ij}$. In order to learn a sparse S, `SLIM` introduces the ℓ_1-norm of S as a regularizer in Equation 3. It is well known that ℓ_1-norm regularization introduces sparsity into the solutions [15]. The matrix S learned by `SLIM` is referred to

as SLIM's aggregation coefficient matrix. Extensive experiments in [10] have shown that SLIM outperforms the state-of-the-art *top-N* recommendation methods.

5. SLIM WITH SIDE INFORMATION

SLIM provides a general framework in which only the aggregation coefficient matrix S is needed for efficient *top-N* recommendations, and this matrix is learned from the user-item purchase profiles. In this section, we present four different extensions of SLIM that are designed to incorporate side information about the items in order to further improve the qualify of the recommendations.

5.1 Collective SLIM

The first approach assumes that there exist correlations between users' co-purchase behaviors on two items and the similarity of the two items' intrinsic properties encoded in their side information. In order to enforce such correlations, this approach imposes the additional requirement that both the user-item purchase profile matrix M and the item side information matrix F should be reproduced by the same sparse linear aggregation. That is, in addition to satisfying $M \sim MS$, the coefficient matrix S should also satisfy

$$F \sim FS. \tag{4}$$

This is achieved by learning the aggregation coefficient matrix S as the minimizer to the following optimization problem:

$$\begin{aligned} \underset{S}{\text{minimize}} \quad & \frac{1}{2}\|M - MS\|_F^2 + \frac{\alpha}{2}\|F - FS\|_F^2 \\ & + \frac{\beta}{2}\|S\|_F^2 + \lambda\|S\|_1 \\ \text{subject to} \quad & S \geq 0, \\ & \text{diag}(S) = 0, \end{aligned} \tag{5}$$

where $\|F - FS\|_F^2$ measures how well the aggregation coefficient matrix S fits the side information. The parameter α is used to control the relative importance of the user-item purchase information M and the item side information F when they are used to learn S. Note that in this method, the side information is actually used to regularize the original SLIM method (i.e., via adding the regularization term $\frac{\alpha}{2}\|F - FS\|_F^2$ into Equation 3). The recommendations are generated in exactly the same way as in SLIM. That is, the recommendation score for user u_i on item t_j is calculated as $\tilde{m}_{ij} = \mathbf{m}_i^\mathsf{T}\mathbf{s}_j$. Since the matrix S is learned from both M and F collectively by using F to regularize the original SLIM method, this approach is referred to as collective SLIM and denoted by cSLIM.

The solution to the optimization problem in Equation 5 is identical to the solution of an optimization problem in the same form as in Equation 3 with M in Equation 3 replaced by $M' = [M, \sqrt{\alpha}F]^\mathsf{T}$.

5.2 Relaxed cSLIM

The second approach also tries to reproduce the item side information using a sparse linear method as in cSLIM, but it uses an alternative approach for achieving this. Specifically, it uses an aggregation coefficient matrix Q to reproduce F as

$$F \sim FQ, \tag{6}$$

where Q is not necessarily identical to S as in Equation 2. Thus, this method is a relaxation from cSLIM. However, the two aggregation coefficient matrices S and Q are tied by requiring that Q should not be significantly different from S (i.e., $S \sim Q$). The matrix S and the matrix Q in Equation 6 are learned as the minimizers of the following optimization problem:

$$\begin{aligned} \underset{S,Q}{\text{minimize}} \quad & \frac{1}{2}\|M - MS\|_F^2 + \frac{\alpha}{2}\|F - FQ\|_F^2 \\ & + \frac{\beta_1}{2}\|S - Q\|_F^2 + \frac{\beta_2}{2}(\|S\|_F^2 + \|Q\|_F^2) \\ & + \lambda(\|S\|_1 + \|Q\|_1) \\ \text{subject to} \quad & S \geq 0, Q \geq 0, \\ & \text{diag}(S) = 0, \text{diag}(Q) = 0, \end{aligned} \tag{7}$$

where the parameter β_1 controls how much S and Q are allowed to be different from each other. Similar to cSLIM, this method regularizes the original SLIM using item side information by adding the two regularization terms $\frac{\alpha}{2}\|F - FQ\|_F^2$ and $\frac{\beta_1}{2}\|S - Q\|_F^2$ and the recommendations are generated in the same way as in SLIM. Since this method is a relaxation from cSLIM, it is refereed to as relaxed collective SLIM and denoted by rcSLIM.

The optimization problem in Equation 7 can be solved via an approach alternating on solving S and Q. In each iteration, one variable is fixed and the problem becomes a regularized optimization problem with respect to the other variable, and it can be solved using a similar approach of stacking matrices as in Section 5.1. The solution of cSLIM is used as the initial value of S.

5.3 Side Information Induced SLIM

An alternative way to learn the aggregation coefficient matrix S of SLIM is to represent S as a function in the item feature space and thus it captures the feature-based relations of the items. One option of achieving this is to use the item-item similarity matrix calculated as FF^T, that is, the aggregation coefficient from one item to another is calculated as the dot-product of their feature vectors (i.e., item-item feature similarity). However, in this way, the aggregation coefficient matrix is not customized to the user-item purchase profiles M at all, and thus a SLIM with such aggregation coefficient matrix can fit M very poorly. Another way is to learn a weighting matrix W such that the aggregation coefficient value s_{ij} can be represented as a linear combination of item t_i's feature $\mathbf{f}_i$ weighted by item t_j's personalized weighting vector $\mathbf{w}_j$ over individual item features, that is, $s_{ij} = \mathbf{f}_i^\mathsf{T}\mathbf{w}_j$ and $\mathbf{w}_j$ is W's j-th column. In this way, the coefficient matrix S can be represented as a weighted linear combination of the item features F using W, that is,

$$S = F^\mathsf{T}W. \tag{8}$$

Such weighting matrix W can be learned as the minimizer of the following optimization problem:

$$\begin{aligned} \underset{W}{\text{minimize}} \quad & \frac{1}{2}\|M - M(F^\mathsf{T}W - D)\|_F^2 \\ & + \frac{\beta}{2}\|W\|_F^2 + \lambda\|F^\mathsf{T}W\|_1 \\ \text{subject to} \quad & W \geq 0 \\ & D = \text{diag}(\text{diag}(F^\mathsf{T}W)), \end{aligned} \tag{9}$$

where $D = \text{diag}(\text{diag}(F^\mathsf{T}W))$ is a diagonal matrix with the corresponding diagonal values from $F^\mathsf{T}W$. D is subtracted from $F^\mathsf{T}W$ so as to ensure that m_{ij} is not used to compute $\tilde{m}_{ij}$, and this is equivalent to the constraint $\text{diag}(S) = 0$ in Equation 3. In this method, the recommendation score for user u_i on item t_j is calculated as $\tilde{m}_{ij} = \mathbf{m}_i^\mathsf{T}(F^\mathsf{T}\mathbf{w}_j - \mathbf{d}_j)$, where $\mathbf{w}_j$ and $\mathbf{d}_j$ is the j-th column of W and D, respectively. Since this method explicitly specifies the aggregation coefficient matrix S as a function of the item side information F, it is referred to as side information induced `SLIM` and denoted by `fSLIM`.

The optimal solution to the optimization problem in Equation 9 is $W^* = [\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_j, \cdots, \mathbf{w}_n]$, where $\mathbf{w}_j$ is the optimal solution to the following problem:

$$\begin{aligned} \underset{\mathbf{w}_j}{\text{minimize}} \quad & \frac{1}{2}\|\mathbf{m}_j - MF_{-j}^\mathsf{T}\mathbf{w}_j\|_F^2 + \frac{\beta}{2}\|\mathbf{w}_j\|_F^2 + \lambda\|\mathbf{c}\mathbf{w}_j\|_1 \\ \text{subject to} \quad & \mathbf{w}_j \geq 0, \end{aligned}$$

where $c_p = \sum_{k=1}^{n} f_{pk}$, and F_{-j} is a matrix with F's j-th column set to 0.

5.4 Side Information Induced Double SLIM

`SLIM` and `fSLIM` have their own advantages. `SLIM` learns the aggregation coefficient matrix S purely from purchase profiles such that it better fits the user-item purchase information. `fSLIM` forces the aggregation coefficient matrix S to be expressed in the item feature space and therefore it captures useful information from the item features. `SLIM` and `fSLIM` can be coupled within one method so as to leverage both their advantages and better learn from purchase profiles and side information concurrently. One way to combine `SLIM` and `fSLIM` is to have the user-item purchase profile M reproduced by both `SLIM` and `fSLIM` as

$$M \sim MS + M(F^\mathsf{T}W - D), \tag{10}$$

where the S and W matrices can be learned as the minimizers of the following optimization problem:

$$\begin{aligned} \underset{S,W}{\text{minimize}} \quad & \frac{1}{2}\|M - MS - M(F^\mathsf{T}W - D)\|_F^2 \\ & + \frac{\beta}{2}(\|S\|_F^2 + \|W\|_F^2) + \lambda(\|S\|_1 + \|F^\mathsf{T}W\|_1) \quad (11) \\ \text{subject to} \quad & S \geq 0, W \geq 0, \\ & \text{diag}(S) = 0, D = \text{diag}(\text{diag}(F^\mathsf{T}W)). \end{aligned}$$

In this method, the recommendation score for user u_i on item t_j is calculated as $\tilde{m}_{ij} = \mathbf{m}_i^\mathsf{T}\mathbf{s}_j + \mathbf{m}_i^\mathsf{T}(F^\mathsf{T}\mathbf{w}_j - \mathbf{d}_j)$, where $\mathbf{w}_j$ and $\mathbf{d}_j$ is the j-th column of W and D, respectively. This method is a combination of `SLIM` and `fSLIM` and thus it is refereed as side information reduced `double` `SLIM` and denoted by `f2SLIM`.

That the optimal solution of W in the problem in Equation 11 is identical to the first l rows of the optimal solution W' to the problem in Equation 9 with F replaced by $[F, I]^\mathsf{T}$ where I is an $n \times n$ identity matrix, and $D' = \text{diag}((F')^\mathsf{T}W')$, whereas the optimal S is the last n rows of W'.

6. EXPERIMENTAL METHODOLOGY

6.1 Datasets

We evaluated the performance of different methods on the following real datasets: `ML100K`, `NF`, `CrossRef`, `Lib`, `BBY`, and `Yelp`, whose characteristics are summarized in Table 1.

ML100K The `ML100K` dataset corresponds to movie ratings and was obtained from the MovieLens research project. The movie plots were fetched from the IMDb database and the words that appear in at least 5 plots are used as the movie side information.

NF The `NF` dataset is a subset extracted from the Netflix Prize dataset. The item side information was generated as in the `ML100K` dataset. Only the movies hat were rated by 10-30 users were selected.

CrossRef The `CrossRef` dataset was obtained from crossref.org, and contains scientific articles and lists of article citations. All the articles (i.e., references) that have DOI links and are cited by at least 50 other articles were first selected. Then the articles which cite more than 3 of such references were selected. In this way, an article-reference dataset is constructed, in which the articles (the references) are analogous to the users (the items). The words in the reference titles are used as side information. The *top-N* recommendation on `CrossRef` dataset becomes a task to recommend a reference for a certain article.

Lib The `Lib` dataset was obtained from the University of Minnesota libraries, and contains the library users and their viewed articles. From the entire library records, the users who viewed at least 5 different articles and the articles that were viewed by at least 10 users were collectively selected to construct an user-article matrix. The words in the article titles are used as the article side information. The *top-N* recommendation on `Lib` is to recommend an article to a user.

BBY The `BBY` dataset is a subset of the BestBuy user-product rating and review dataset from BestBuy website (https://developer.bestbuy.com/documentation/archives). The products that were reviewed by at least 5 users and the users who reviewed at least 2 such products were collectively selected so as to construct the dataset. The side information for each item was the text of all the reviews of that item.

Yelp The `Yelp` dataset is a subset of the academic version of Yelp user-business rating and review dataset downloaded from Yelp (http://www.yelp.com/academic_dataset). The users who reviewed at least 3 businesses and the corresponding businesses were selected to construct the dataset. The side information for each item was constructed from the reviews in a way similar to the `BBY` dataset.

For the original rating datasets (i.e., `ML100K`, `NF`, `BBY`, `Yelp`), the multivariate rating values were converted to 1's.

6.2 Evaluation Methodology & Metrics

We applied 5-time Leave-One-Out cross validation to evaluate the performance of different methods. In each run, each of the datasets is split into a training set and a testing set by randomly selecting one of the non-zero entries of each user and placing it into the testing set. The evaluation is conducted by comparing the size-N (by default $N = 10$) recommendation list for each user and the item of that user in the testing set.

The recommendation quality is measured by the Hit Rate (ZR) and the Average Reciprocal Hit-Rank (ARHR) [4]. ZR is defined as follows,

$$\text{HR} = \frac{\#\text{hits}}{\#\text{users}}, \tag{12}$$

where #users is the total number of users, and #hits is the number of users whose item in the testing set is recommended (i.e., hit) in the size-N recommendation list. A

Table 1: The Datasets Used in Evaluation

dataset	purchase information						side information					
	#users	#items	#nnzs	rsize	csize	density	desc	#ftr	#nnz	srsize	scsize	sdensity
ML100K	943	1,682	100,000	106.0	59.5	6.30%	plots	2,327	46,915	27.9	20.2	1.20%
NF	3,086	6,909	128,134	41.5	18.6	0.60%	plots	5,941	200,148	29.0	33.7	0.49%
CrossRef	84,260	23,458	466,068	5.5	19.9	0.02%	titles	5,677	149,839	6.4	26.4	0.11%
Lib	13,843	12,123	103,428	7.47	8.53	0.06%	titles	9,991	86,065	7.1	8.6	0.07%
BBY	127,285	7,330	162,451	1.3	22.2	0.02%	reviews	9,686	1,912,444	260.9	197.4	2.7%
Yelp	13,574	6,896	89,608	6.6	13.0	0.10%	reviews	10,305	330,865	48.0	32.1	0.47%

Columns corresponding to purchase information and side information show the dataset statistics for historical profile matrix M and side information matrix F, respectively. Under purchase information, column corresponding to #users, #items and #nnzs show the number of users, items and non-zero values in each dataset, respectively. Column corresponding to rsize, csize and density shows the average row density, the average column density and the matrix density, respectively. Under side information, column corresponding to desc shows the side information types. Column corresponding to #ftr and #nnz show the dimensionality of side information and the number of non-zero values in the side information, respectively. Column corresponding to srsize, scsize and sdensity show the average row density, the average column density and the density of the side information matrix, respectively.

second measure for evaluation is ARHR, which is defined as follows:

$$\text{ARHR} = \frac{1}{\#\text{users}} \sum_{i=1}^{\#\text{hits}} \frac{1}{p_i}, \tag{13}$$

where if an item of a user is hit, p is the position of the item in the ranked recommendation list. ARHR is a weighted version of HR and it measures how strongly an item is recommended, in which the weight is the reciprocal of the hit position in the recommendation list.

6.3 Side Information Representation

Besides the learning capability of the SSLIM methods, the representation of the side information can impact the overall performance. Since the side information in our datasets is text (e.g., movie plots, product reviews, etc), we investigated different text representations. In all of these schemes, the text of the side information was preprocessed to eliminate stop words and each word was converted to its stem[1].

Binary Representation ($\mathsf{F_b}$) In this scheme, the text of the side information is represented using the bag-of-words model, and the frequency of each word is set to one. The reason for the binarization is that typically the text for an item is short, and there are not many informative words occurring multiple times, and thus a binarized vector is almost same as the original count vector. In addition, since the user-item profile M is binary, intuitively the item-item coefficient matrix Q learned from a binary feature matrix F should be comparable to the aggregation coefficient matrix S learned from M in terms of the values. In this case, the regularization using Q on S (i.e., the $\frac{\beta_1}{2}\|S-Q\|_F^2$ term in rcSLIM) can be more effective.

Normalized TFIDF Representation ($\mathsf{F_{tfidf}}$) For the methods that directly learn from the item text (fSLIM and f2SLIM), it is essential that the text presentation encodes how important a word is in the text. For this purpose, a normalized TFIDF scheme is adopted. The TFIDF scheme [13] is widely used for weighting words in text mining. After the TFIDF scheme is applied on the feature vectors, the feature vectors are normalized to unit length.

Normalized TFIDF Representation with Feature Selection ($\mathsf{F_{tfidf\_fs}}$) Another representation scheme is a modification of $\mathsf{F_{tfidf}}$ by using feature selection. For each feature vector, the words were sorted in decreasing order according to their weights in the TFIDF representation. Then the highest weighted words were selected until cumulatively they contribute to 90% of the vector length.

[1]http://glaros.dtc.umn.edu/gkhome/fetch/sw/cluto/doc2mat-1.0.tar.gz

Normalized TFIDF Binary Representation with Feature Selection ($\mathsf{F_{tfidf\_fs\_b}}$) The last side information representation scheme is a compromise of $\mathsf{F_b}$ and $\mathsf{F_{tfidf\_fs}}$, that is, it converts all the values that are calculated from $\mathsf{F_{tfidf\_fs}}$ to binary. This scheme tries to use only the words that are considered as important by $\mathsf{F_{tfidf\_fs}}$ and meanwhile still retain the advantages of the binary representations.

6.4 Comparison Methods

Singh *et al* [14] proposed the collective matrix factorization (CMF) method for relational learning as follow:

$$\begin{aligned}\underset{U,V,W}{\text{minimize}} \quad & \frac{1}{2}\|M-UV\|_F^2 + \frac{\alpha}{2}\|F-WV\|_F^2 \\ & + \frac{\beta}{2}(\|U\|_F^2 + \|V\|_F^2 + \|W\|_F^2),\end{aligned} \tag{14}$$

where U is an $m \times k$ user factor from M, W is an $l \times k$ feature factor from F, and V is an $k \times n$ item factor which is collectively learned from both M and F. Particularly, $k \ll \min(m,n,l)$. CMF and cSLIM are similar in the sense that a common matrix is learned from both M and F concurrently. However, they are fundamentally different methods. The cSLIM method conforms to linear methods and it models the *top-N* recommendation process as an aggregation on items. On the contrary, CMF models the recommendation process in a low-dimension latent space.

Thu *et al* [7] proposed a weighted regularized matrix factorization (WRMF) method for *top-N* recommendation, which weights the purchase and nonpurchase activities in M differently using a weighting matrix C as follows:

$$\underset{U,V}{\text{minimize}} \quad \frac{1}{2}\|C \circ (M-UV)\|_F^2 + \frac{\beta}{2}(\|U\|_F^2 + \|V\|_F^2). \tag{15}$$

Inspired by this weighting method, we combined WRMF with CMF so as to have a collective weighted regularized matrix factorization method, denoted by CWRMF, as follows:

$$\begin{aligned}\underset{U,V,W}{\text{minimize}} \quad & \frac{1}{2}\|C \circ (M-UV)\|_F^2 + \frac{\alpha}{2}\|F-WV\|_F^2 \\ & + \frac{\beta_1}{2}\|U\|_F^2 + \frac{\lambda}{2}\|V\|_F^2 + \frac{\beta_2}{2}\|W\|_F^2,\end{aligned} \tag{16}$$

in which M and F are still collectively factorized but errors from M are weighted differently by C. We use WRMF and CWRMF as the comparison algorithms in the experiments. In addition, we use another two collaborative filtering methods for comparison purposes. The itemkNN method is a widely used item-based collaborative filtering method proposed in [4]. The itemSI method is a modification of itemkNN, in which the item similarities are calculated as a linear combination of the similarity values calculated from itemkNN and

Table 3: Performance Improvement over SLIM

feature	itemSI	CWRMF	cSLIM	rcSLIM	fSLIM	f2SLIM
F_b	0.973	0.674	1.095	1.048	0.818	1.008
F_{tfidf}	0.988	0.674	1.090	1.062	0.877	1.026
F_{tfidf_fs}	0.974	0.707	1.090	1.063	0.873	1.027
$F_{tfidf\_fs\_b}$	0.970	0.707	1.113	1.069	0.828	1.012
avg	0.976	0.690	1.097	1.061	0.849	1.018

Each value in the first four rows is calculated as the geometric mean of HR ratios of the corresponding method over SLIM over all the datasets, given the corresponding feature representation scheme used. The values in the last row is calculated as the geometric mean of HR ratios of the corresponding method over SLIM over all the datasets and all the feature presentation schemes.

the cosine similarity values calculated from side information weighted by a parameter α.

7. RESULTS

7.1 Overall Performance

Table 2 presents the detailed results of the SSLIM methods (cSLIM, rcSLIM, fSLIM and f2SLIM), the three methods without side information (itemkNN, WRMF and SLIM) and another two methods that utilize side information (itemSI and CWRMF), with respect to different side information representation schemes (F_b, F_{tfidf}, F_{tfidf_fs} and $F_{tfidf\_fs\_b}$). For the methods itemkNN, WRMF andSLIM, F_{no} is used in Table 2 to denote that side information is not used.

Table 2 shows that SLIM outperforms the other methods that do not utilize side information (i.e., itemkNN and WRMF) on all the datasets except BBY. For the BBY dataset, WRMF performs the best. This conforms to the conclusions as in [10], and thus we use SLIM as the baseline to further evaluate all the methods that utilize side information.

Table 3 summarizes the overall performance of the different methods that utilize side information, with respect to SLIM. Irrespective of the feature representation scheme, cSLIM, rcSLIM and f2SLIM perform better than SLIM with average improvement 9.7%, 6.1% and 1.8%, respectively (the last row in Table 3). This demonstrates that side information contains useful information, and proper incorporation of side information into the recommender systems can bring significant performance improvement.

The methods itemSI, fSLIM and CWRMF perform worse than SLIM. The itemSI method is a trivial extension of itemkNN and it does not involve any learning. fSLIM is a method that learns directly from the side information. The performance of fSLIM indicates that this method may not be able to pick out and highly weight the individual features in the item side information that are most relevant to the recommendations. CWRMF is the worst one and even worse than itemSI. This may be related to the discussion on CMF as in Agarwal *et al* [2], that is, when the side information is sparse, CMF may not work well.

Comparing the gains that can be obtained by utilizing side information across the different datasets, we see that they are not uniform. For the two movie datasets (ML100K and NF), the side information provides minimal benefits, whereas the gains achieved from the other datasets is substantial. We believe that this is due to the fact that the side information used for the movie datasets was quite generic, and does not contain sufficient information.

7.2 Side Information Representation

Table 3 shows that the performance of the side information representation schemes depends on the recommendation methods. For cSLIM, which uses side information for regularization, the binary feature representations (F_b and $F_{tfidf\_fs\_b}$) lead to better performance than the multivariate feature representations (F_{tfidf} and F_{tfidf_fs}). This may be due to the fact that the binary features are treated homogeneously as the user-item purchase data and thus they can regularize the learning process effectively. However, for the methods fSLIM and f2SLIM, which involve direct learning from side information, F_{tfidf} and F_{tfidf_fs} result in better performance than the binary ones, since they differentiate the importance of features within the representations. In general, fSLIM and f2SLIM prefer the feature representations that encode word importance, so even the binary representation $F_{tfidf\_fs\_b}$, which has feature selection applied, also outperforms F_b, which does not differentiate features at all.

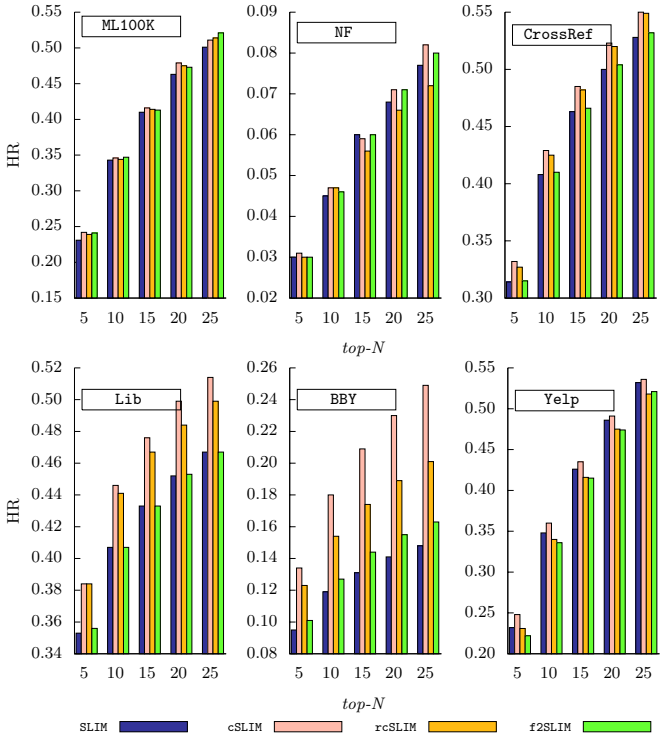

Figure 1: Recommendations for Different N Values

7.3 Recommendation on Different Top-N

Figure 1 presents the performance of SLIM and SSLIM methods (except fSLIM since it performs poorly) for *top-N* recommendation with different values of N. The $F_{tfidf\_fs\_b}$ side information representation is used for all the methods. For all the datasets, cSLIM consistently outperforms SLIM and other SSLIM methods on all N values (except $N = 25$ for dataset ML100K and $N = 15$ for dataset NF). The rcSLIM method is the second competitive methods over all N values. The f2SLIM method shows performance that is comparable to SLIM and in some cases (i.e., $N = 25$ for ML100K, all the N values for BBY) it outperforms SLIM. It performs consistently worse than SLIM only on Yelp.

7.4 Density Studies on the Purchase Data

To understand how the density of the dataset impacts the gains that can be obtained by utilizing side information, we

Table 2: Performance Comparison on All the Datasets

feature	method	ML100K params				HR	ARHR	NF params				HR	ARHR	CrossRef params				HR	ARHR
F_{no}	itemkNN	10	-	-	-	0.287	0.124	300	-	-	-	0.043	0.015	300	-	-	-	0.404	0.212
	WRMF	5	50	1	-	0.327	0.133	5	400	2	-	0.045	0.017	1	400	10	-	0.352	0.169
	SLIM	2	2	-	-	0.343	0.147	5	0.5	-	-	0.045	0.018	5	0.5	-	-	0.408	0.211
F_{b}	itemSI	200	0.5	-	-	0.262	0.116	200	0.5	-	-	0.042	0.016	200	0.5	-	-	0.416	0.218
	CWRMF	1	0.5	50	1	0.327	0.139	1	0.5	200	2	0.028	0.009	1	2	200	2	0.168	0.080
	cSLIM	0.5	10	1	-	0.344	0.148	0.1	10	1	-	0.047	0.017	1	5	0.1	-	0.429	0.224
	rcSLIM	0.1	5	1	2	0.344	0.147	5	10	10	1	0.047	0.017	5	10	5	0.5	0.421	0.218
	fSLIM	10	0.5	-	-	0.303	0.130	10	0.1	-	-	0.030	0.011	2	0.1	-	-	0.287	0.139
	f2SLIM	10	1	-	-	0.343	0.146	5	0.1	-	-	0.046	0.017	5	0.1	-	-	0.410	0.211
F_{tfidf}	itemSI	200	0.1	-	-	0.261	0.120	300	0.5	-	-	0.043	0.016	200	0.5	-	-	0.416	0.218
	CWRMF	1	0.5	50	1	0.327	0.139	1	0.5	200	2	0.028	0.009	1	2	200	2	0.168	0.080
	cSLIM	2	5	5	-	0.344	0.147	0.5	10	1	-	0.047	0.017	2	5	0.1	-	0.426	0.220
	rcSLIM	1	2	2	2	0.345	0.147	0.5	10	2	1	0.047	0.018	5	10	5	0.1	0.422	0.217
	fSLIM	2	0.1	-	-	0.317	0.134	5	0.1	-	-	0.035	0.011	2	0.1	-	-	0.292	0.141
	f2SLIM	5	5	-	-	0.341	0.147	10	0.5	-	-	0.046	0.017	5	0.1	-	-	0.413	0.212
F_{tfidf_fs}	itemSI	200	0.5	-	-	0.260	0.118	50	0.5	-	-	0.042	0.015	200	0.5	-	-	0.416	0.218
	CWRMF	1	1	50	1	0.325	0.125	1	2	200	1	0.031	0.010	1	2	200	2	0.174	0.082
	cSLIM	2	5	5	-	0.344	0.147	1	10	1	-	0.047	0.017	2	5	0.1	-	0.425	0.220
	rcSLIM	1	2	2	2	0.345	0.147	1	10	10	0.1	0.047	0.017	5	10	5	0.1	0.422	0.217
	fSLIM	2	0.1	-	-	0.317	0.133	5	0.1	-	-	0.034	0.012	2	0.1	-	-	0.292	0.141
	f2SLIM	5	5	-	-	0.343	0.148	10	5	-	-	0.046	0.017	5	0.1	-	-	0.413	0.212
$F_{tfidf\_fs\_b}$	itemSI	200	0.1	-	-	0.262	0.121	50	0.5	-	-	0.042	0.015	200	0.5	-	-	0.416	0.218
	CWRMF	1	1	50	1	0.325	0.125	1	2	200	1	0.031	0.010	1	2	200	2	0.174	0.082
	cSLIM	0.5	5	2	-	0.346	0.151	0.5	5	0.5	-	0.047	0.018	1	5	0.1	-	0.429	0.223
	rcSLIM	5	10	10	0.1	0.344	0.149	5	10	2	1	0.047	0.019	5	10	5	0.1	0.425	0.220
	fSLIM	20	1	-	-	0.312	0.134	10	0.1	-	-	0.029	0.011	2	0.1	-	-	0.287	0.139
	f2SLIM	10	1	-	-	0.347	0.151	10	1	-	-	0.046	0.017	5	0.1	-	-	0.410	0.211

feature	method	Lib params				HR	ARHR	BBY params				HR	ARHR	Yelp params				HR	ARHR
F_{no}	itemkNN	10	-	-	-	0.385	0.247	200	-	-	-	0.070	0.032	100	-	-	-	0.257	0.092
	WRMF	0.5	200	10	-	0.361	0.221	0.5	100	10	-	0.124	0.062	5	50	10	-	0.327	0.131
	SLIM	5	0.5	-	-	0.407	0.266	10	0.1	-	-	0.119	0.064	20	0.1	-	-	0.348	0.147
F_{b}	itemSI	300	0.5	-	-	0.407	0.258	100	2	-	-	0.151	0.068	300	1	-	-	0.321	0.127
	CWRMF	1	2	200	2	0.278	0.162	1	0.5	50	2	0.095	0.041	1	0.5	50	2	0.244	0.100
	cSLIM	1	5	0.1	-	0.443	0.283	0.1	20	0.1	-	0.165	0.081	0.1	20	0.5	-	0.361	0.154
	rcSLIM	5	10	2	0.5	0.439	0.288	5	10	10	0.5	0.139	0.079	2	10	10	0.5	0.338	0.144
	fSLIM	2	0.1	-	-	0.358	0.233	10	0.1	-	-	0.107	0.055	10	0.1	-	-	0.319	0.137
	f2SLIM	5	0.5	-	-	0.408	0.267	10	0.1	-	-	0.127	0.068	10	0.5	-	-	0.332	0.138
F_{tfidf}	itemSI	300	0.5	-	-	0.407	0.258	100	2	-	-	0.163	0.072	200	1	-	-	0.318	0.127
	CWRMF	1	2	200	2	0.278	0.162	1	0.5	100	1	0.095	0.041	1	0.5	50	2	0.244	0.100
	cSLIM	2	5	0.1	-	0.436	0.280	2	50	0.1	-	0.169	0.087	2	50	0.1	-	0.351	0.147
	rcSLIM	5	10	2	0.1	0.434	0.286	5	10	10	0.1	0.152	0.081	5	10	10	0.1	0.338	0.143
	fSLIM	2	0.1	-	-	0.367	0.236	2	0.1	-	-	0.115	0.059	10	0.1	-	-	0.353	0.148
	f2SLIM	5	0.5	-	-	0.410	0.266	20	0.1	-	-	0.132	0.069	20	0.1	-	-	0.354	0.148
F_{tfidf_fs}	itemSI	300	0.5	-	-	0.408	0.257	200	2	-	-	0.159	0.069	300	1	-	-	0.308	0.120
	CWRMF	1	2	200	2	0.286	0.167	1	2	100	1	0.102	0.044	1	2	50	2	0.260	0.103
	cSLIM	2	5	0.1	-	0.436	0.280	2	50	0.1	-	0.169	0.087	2	50	0.1	-	0.351	0.147
	rcSLIM	5	10	1	0.5	0.435	0.286	5	10	10	0.1	0.152	0.081	5	10	10	0.1	0.338	0.143
	fSLIM	2	0.1	-	-	0.367	0.236	2	0.1	-	-	0.115	0.059	10	0.1	-	-	0.353	0.148
	f2SLIM	5	0.5	-	-	0.410	0.267	20	0.1	-	-	0.132	0.069	20	0.1	-	-	0.354	0.148
$F_{tfidf\_fs\_b}$	itemSI	300	0.5	-	-	0.407	0.257	100	2	-	-	0.153	0.068	300	1	-	-	0.311	0.120
	CWRMF	1	2	200	2	0.286	0.167	1	2	100	1	0.102	0.044	1	2	50	2	0.260	0.103
	cSLIM	1	5	0.1	-	0.446	0.284	0.1	50	0.1	-	0.180	0.090	0.5	20	0.1	-	0.360	0.156
	rcSLIM	5	10	2	0.1	0.441	0.287	5	10	10	0.1	0.154	0.085	2	10	10	0.1	0.340	0.144
	fSLIM	2	0.1	-	-	0.358	0.234	20	0.1	-	-	0.109	0.057	20	0.1	-	-	0.338	0.141
	f2SLIM	5	0.5	-	-	0.407	0.267	10	0.1	-	-	0.127	0.069	20	0.1	-	-	0.337	0.139

The parameters represented for each method are as follows, respectively: `itemkNN` and `itemSI`: the number of neighbors k, and weighting parameter α; `SLIM`: the ℓ_2-norm regularization parameter β and the ℓ_1-norm regularization parameter λ as in [10]; `cSLIM` and `rcSLIM`: the weighting parameter on side information regularizer α, the ℓ_2-norm regularization parameter β (β_1 and β_2) and the ℓ_1-norm regularization parameter λ as in Equation 5 and 7; `fSLIM` and `f2SLIM`: the ℓ_2-norm regularization parameter β and the ℓ_1-norm regularization parameter λ as in Equation 9 and 11; `WRMF`: the regularization parameter β, the number of latent dimensions k and the weight on non-zero values c as in [7]; `CWRMF`: the weighting parameter on side information α, the regularization parameter λ, the number of latent dimensions k and the weight on non-zero values c as in Equation 15; Columns corresponding to HR and ARHR present the hit rate and average reciprocal hit-rank, respectively. N in this table is 10.

performed a series of experiments in which we removed some of the user-item purchase profile data as follows. For each dataset, we always keep the testing set and side information unchanged, but randomly select a certain percentage (defined as density factor) of non-zero values from each user so as to construct training sets of different information density. Figure 2 presents the results from different recommendation methods on the datasets. The results in these figures are relative to the performance achieved by `SLIM` at the same density level.

These results show that in general, `cSLIM` and `rcSLIM` lead to more significant performance gains when the user-item purchase profiles are sparser. This indicates that when the user-item purchase data is sparse, `cSLIM` and `rcSLIM` are more effective in exploiting and incorporating side information and lead to more accurate *top-N* recommendations. Note that for the dataset `ML100K` and `NF`, performance improvement from incorporating side information is hardly ob-

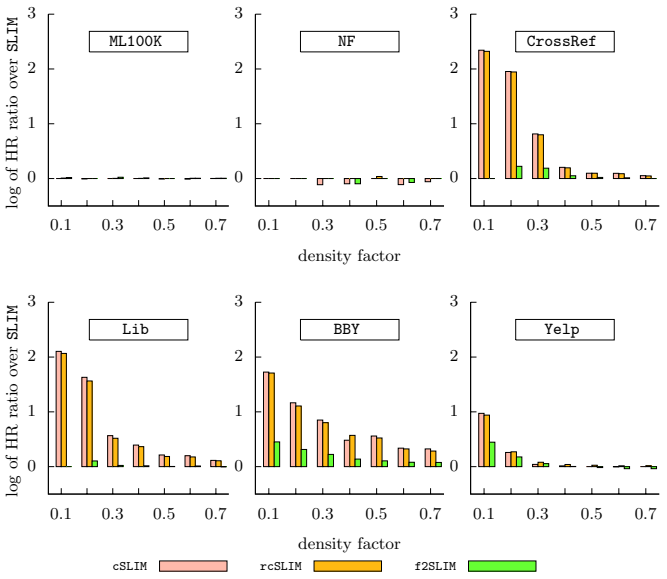

Figure 2: Density Studies

served. As discussed in Section 7.1, we believe that this is due to the low quality of the side information.

8. CONCLUSIONS

This paper focused on incorporating side information into the sparse linear methods (SLIM) for *top-N* recommender systems. We developed four different approaches that incorporate side information during the estimation of SLIM's aggregation coefficient matrix. Our experiments showed that the developed methods lead to measurable improvements over the original SLIM methods that relied solely on user-item purchase profiles.

Acknowledgement

This work was supported in part by NSF (IIS-0905220, OCI-1048018, and IOS-0820730), the DOE grant USDOE/DE-SC0005013 and the Digital Technology Center at the University of Minnesota. Access to research and computing facilities was provided by the Digital Technology Center and the Minnesota Supercomputing Institute.

9. REFERENCES

[1] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining*, pages 19–28, 2009.

[2] D. Agarwal, B.-C. Chen, and B. Long. Localized factor models for multi-context recommendation. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 609–617, New York, NY, USA, 2011. ACM.

[3] L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, and M. A. Rueda-Morales. Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks. *International Journal of Approximate Reasoning*, 51(7):785 – 799, 2010.

[4] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems*, 22:143–177, January 2004.

[5] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. *IEEE International Conference on Data Mining*, pages 176–185, 2010.

[6] A. Gunawardana and C. Meek. A unified approach to building hybrid recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, RecSys '09, pages 117–124, New York, NY, USA, 2009. ACM.

[7] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 263–272, Washington, DC, USA, 2008. IEEE Computer Society.

[8] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 79–86, 2010.

[9] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 426–434, New York, NY, USA, 2008. ACM.

[10] X. Ning and G. Karypis. Slim: Sparse linear methods for top-n recommender systems. In *Proceedings of 11th IEEE International Conference on Data Mining*, pages 497–506, 2011.

[11] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information*, SIGIR '11, pages 635–644, New York, NY, USA, 2011. ACM.

[12] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors. *Recommender Systems Handbook*. Springer, 2011.

[13] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[14] A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. In *Proceeding of the 14th ACM International Conference on Knowledge Discovery and Data Mining*, pages 650–658, 2008.

[15] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288, 1996.

[16] S.-H. Yang, B. Long, A. Smola, N. Sadagopan, Z. Zheng, and H. Zha. Like like alike: joint friendship and interest propagation in social networks. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 537–546, New York, NY, USA, 2011. ACM.