

# Top-N Recommender System via Matrix Completion

Zhao Kang, Chong Peng, Qiang Cheng

Department of Computer Science,  
Southern Illinois University,  
Carbondale, IL 62901, USA  
{Zhao.Kang, pchong, qcheng}@siu.edu

## Abstract

Top-N recommender systems have been investigated widely both in industry and academia. However, the recommendation quality is far from satisfactory. In this paper, we propose a simple yet promising algorithm. We fill the user-item matrix based on a low-rank assumption and simultaneously keep the original information. To do that, a nonconvex rank relaxation rather than the nuclear norm is adopted to provide a better rank approximation and an efficient optimization strategy is designed. A comprehensive set of experiments on real datasets demonstrates that our method pushes the accuracy of Top-N recommendation to a new level.

## Introduction

The growth of online markets has made it increasingly difficult for people to find items which are interesting and useful to them. Top-N recommender systems have been widely adopted by the majority of e-commerce web sites to recommend size- $N$  ranked lists of items that best fit customers' personal tastes and special needs (Linden, Smith, and York 2003). It works by estimating a consumer's response for new items, based on historical information, and suggesting to the consumer novel items for which the predicted response is high. In general, historical information can be obtained explicitly, for example, through ratings/reviews, or implicitly, from purchase history or access patterns (Desrosiers and Karypis 2011).

Over the past decade, a variety of approaches have been proposed for Top-N recommender systems (Ricci, Rokach, and Shapira 2011). They can be roughly divided into three categories: neighborhood-based collaborative filtering, model-based collaborative filtering, and ranking-based methods. The general principle of neighborhood-based methods is to identify the similarities among users/items (Deshpande and Karypis 2004). For example, item-based k-nearest-neighbor (ItemKNN) collaborative filtering methods first identify a set of similar items for each of the items that the consumer has purchased, and then recommend Top-N items based on those similar items. However, they suffer from low accuracy since they employ few item characteristics.

Model-based methods build a model and then generate recommendations. For instance, the widely studied matrix factorization (MF) methods employ the user-item similarities in their latent space to extract the user-item purchase patterns. Pure singular-value-decomposition-based (PureSVD) matrix factorization method (Cremonesi, Koren, and Turrin 2010) characterizes users and items by the most principal singular vectors of the user-item matrix. A weighted regularized matrix factorization (WRMF) (Pan et al. 2008; Hu, Koren, and Volinsky 2008) method applies a weighting matrix to differentiate the contributions from observed purchase/rating activities and unobserved ones.

The third category of methods rely on ranking/retrieval criteria. Here, Top-N recommendation is treated as a ranking problem. A Bayesian personalized ranking (BPR) (Rendle et al. 2009) criterion, which is the maximum posterior estimator from a Bayesian analysis, is used to measure the difference between the rankings of user-purchased items and the rest items. BPR can be combined with ItemKNN (BPRkNN) and MF method (BPRMF). One common drawback of these approaches lies in low recommendation quality.

Recently, a novel Top-N recommendation method SLIM (Ning and Karypis 2011) has been proposed. From user-item matrix  $X$  of size  $m \times n$ , it first learns a sparse aggregation coefficient matrix  $W \in \mathcal{R}_+^{n \times n}$  by encoding each item as a linear combination of all other items and solving an  $l_1$ -norm and  $l_2$ -norm regularized optimization problem. Each entry  $w_{ij}$  describes the similarity between item  $i$  and  $j$ . SLIM has obtained better recommendation accuracy than the other state-of-the-art methods. However, SLIM can only capture relations between items that are co-purchased/co-rated by at least one user, while an intrinsic characteristic of recommender systems is sparsity due to the fact that users typically rate only a small portion of the available items.

To overcome the above limitation, LorSLIM (Cheng, Yin, and Yu 2014) has also imposed a low-rank constraint on  $W$ . It solves the following problem:

$$\begin{aligned} \min_W \quad & \frac{1}{2} \|X - XW\|_F^2 + \frac{\beta}{2} \|W\|_F^2 + \lambda \|W\|_1 + \gamma \|W\|_* \\ \text{s.t.} \quad & W \geq 0, \quad \text{diag}(W) = 0, \end{aligned}$$

where  $\|W\|_*$  is the nuclear norm of  $W$ , defined as the sum of its singular values. Low-rank structure is motivated by the fact that a few latent variables from  $F$  that explain items'

features in factor model  $W \approx FF^T$  are of low rank. After obtaining  $W$ , the recommendation score for user  $i$  about an un-purchased/rated item  $j$  is  $\hat{x}_{ij} = \mathbf{x}_i^T \mathbf{w}_j$ , where  $x_{ij} = 0$ ,  $\mathbf{x}_i^T$  is the  $i$ -th row of  $X$ , and  $\mathbf{w}_j$  is the  $j$ -th column of  $W$ . Thus  $\hat{X} = XW$ . LorSLIM can model the relations between items even on sparse datasets and thus improves the performance.

To further boost the accuracy of Top-N recommender systems, we first fill the missing ratings by solving a nonconvex optimization problem, based on the assumption that the user's ratings are affected by only a few factors and the resulting rating matrix should be of low rank (Lee et al. 2014), and then make the Top-N recommendation. This is different from previous approaches: Middle values of the rating ranges, or the average user or item ratings are commonly utilized to fill out the missing ratings (Breese, Heckerman, and Kadie 1998; Deshpande and Karypis 2004); a more reliable approach utilizes content information (Melville, Mooney, and Nagarajan 2002; Li and Zaiiane 2004; Degemmis, Lops, and Semeraro 2007), for example, the missing ratings are provided by autonomous agents called filterbots (Good et al. 1999), which rate items based on some specific characteristics of their content; a low rank matrix factorization approach seeks to approximate  $X$  by a multiplication of low rank factors (Yu et al. 2009). Experimental results demonstrate the superior recommendation quality of our approach.

Due to the inherent computational complexity of rank problems, the non-convex rank function is often relaxed to its convex surrogate, i.e. the nuclear norm (Candès and Recht 2009; Recht, Xu, and Hassibi 2008). However, this substitution is not always valid and can lead to a biased solution (Shi and Yu 2011; Kang, Peng, and Cheng 2015b). Matrix completion with nuclear norm regularization can be significantly hurt when entries of the matrix are sampled non-uniformly (Srebro and Salakhutdinov 2010). Nonconvex rank approximation has received significant attention (Zhong et al. 2015; Kang and Cheng 2015). Thus we use log-determinant ( $\log\det$ ) function to approximate the rank function and design an effective optimization algorithm.

### Problem Formulation

The incomplete user-item purchases/ratings matrix is denoted as  $M$  of size  $m \times n$ .  $M_{ij}$  is 1 or a positive value if user  $i$  has ever purchased/rated item  $j$ ; otherwise it is 0. Our goal is to reconstruct a full matrix  $X$ , which is supposed to be low-rank. Consider the following matrix completion problem:

$$\begin{aligned} \min_X \log\det((X^T X)^{1/2} + I) \\ \text{s.t. } X_{ij} = M_{ij}, \quad (i, j) \in \Omega, \end{aligned} \quad (1)$$

where  $\Omega$  is the set of locations corresponding to the observed entries and  $I \in \mathcal{R}^{n \times n}$  is an identity matrix. It is easy to show that  $\log\det((X^T X)^{1/2} + I) \leq \|X\|_*$ , i.e.,  $\log\det$  is a tighter rank approximation function than the nuclear norm.  $\log\det$  also helps mitigate another inherent disadvantage of the nuclear norm, i.e., the imbalanced penalization of different singular values (Kang, Peng, and Cheng 2015a). Previously  $\log\det(X + \delta I)$  was suggested to restrict the rank

of positive semidefinite matrix  $X$  (Fazel, Hindi, and Boyd 2003), which is not guaranteed for more general  $X$ , and also  $\delta$  is required to be small, which leads to significantly biased approximation for small singular values. Compared to some other nonconvex relaxations in the literature (Lu et al. 2014), our formulation enjoys the simplicity and efficacy.

### Methodology

Considering that the user-item matrix is often nonnegative, we add nonnegative constraint  $X \geq 0$ , i.e., element-wise positivity, for easy interpretation of the representation. Let  $\mathcal{P}_\Omega$  be the orthogonal projection operator onto the span of matrices vanishing outside of  $\Omega$  (i.e.,  $\Omega^c$ ) so that

$$(\mathcal{P}_\Omega(X))_{ij} = \begin{cases} X_{ij}, & \text{if } X_{ij} \in \Omega; \\ 0, & \text{if } X_{ij} \in \Omega^c. \end{cases}$$

Problem (1) can be reformulated as

$$\begin{aligned} \min_X \log\det((X^T X)^{1/2} + I) + l_{\mathcal{R}_+}(X) \\ \text{s.t. } \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M), \end{aligned} \quad (2)$$

where  $l_{\mathcal{R}_+}$  is the indicator function, defined element-wisely as

$$l_{\mathcal{R}_+}(x) = \begin{cases} 0, & \text{if } x \geq 0; \\ +\infty, & \text{otherwise.} \end{cases}$$

Notice that this is a nonconvex optimization problem, which is not easy to solve in general. Here we develop an effective optimization strategy based on augmented Lagrangian multiplier (ALM) method. By introducing an auxiliary variable  $Y$ , it has the following equivalent form

$$\begin{aligned} \min_{X, Y} \log\det((X^T X)^{1/2} + I) + l_{\mathcal{R}_+}(Y) \\ \text{s.t. } \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M), \quad X = Y, \end{aligned} \quad (3)$$

which has an augmented Lagrangian function of the form

$$\begin{aligned} \mathcal{L}(X, Y, Z) = \log\det((X^T X)^{1/2} + I) + l_{\mathcal{R}_+}(Y) + \\ \frac{\mu}{2} \|X - Y + \frac{Z}{\mu}\|_F^2 \quad \text{s.t. } \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M), \end{aligned} \quad (4)$$

where  $Z$  is a Lagrange multiplier and  $\mu > 0$  is a penalty parameter.

Then, we can apply the alternating minimization idea to update  $X, Y$ , i.e., updating one of the two variables with the other fixed.

Given the current point  $X^t, Y^t, Z^t$ , we update  $X^{t+1}$  by solving

$$\begin{aligned} X^{t+1} = \arg \min_X \log\det((X^T X)^{1/2} + I) + \\ \frac{\mu^t}{2} \|X - Y^t + \frac{Z^t}{\mu^t}\|_F^2 \end{aligned} \quad (5)$$

This can be converted to scalar minimization problems due to the following theorem (Kang et al. 2015).

---

**Algorithm 1** Solve (3)

---

**Input:** Original incomplete data matrix  $M_\Omega \in \mathcal{R}^{m \times n}$ , parameters  $\mu^0 > 0, \gamma > 1$ .

**Initialize:**  $Y = \mathcal{P}_\Omega(M)$ ,  $Z = 0$ .

**REPEAT**

- 1: Obtain  $X$  through (10).
- 2: Update  $Y$  as (12).
- 3: Update the Lagrangian multipliers  $Z$  by

$$Z^{t+1} = Z^t + \mu^t (X^{t+1} - Y^{t+1}).$$

- 4: Update the parameter  $\mu^t$  by  $\mu^{t+1} = \gamma \mu^t$ .

**UNTIL** stopping criterion is met.

---

**Theorem 1** If  $F(Z)$  is a unitarily invariant function and SVD of  $A$  is  $A = U\Sigma_A V^T$ , then the optimal solution to the following problem

$$\min_Z F(Z) + \frac{\beta}{2} \|Z - A\|_F^2 \quad (6)$$

is  $Z^*$  with SVD  $U\Sigma_Z^* V^T$ , where  $\Sigma_Z^* = \text{diag}(\sigma^*)$ ; moreover,  $F(Z) = f \circ \sigma(Z)$ , where  $\sigma(Z)$  is the vector of nonincreasing singular values of  $Z$ , then  $\sigma^*$  is obtained by using the Moreau-Yosida proximity operator  $\sigma^* = \text{prox}_{f,\beta}(\sigma_A)$ , where  $\sigma_A := \text{diag}(\Sigma_A)$ , and

$$\text{prox}_{f,\beta}(\sigma_A) := \arg \min_{\sigma \geq 0} f(\sigma) + \frac{\beta}{2} \|\sigma - \sigma_A\|_2^2. \quad (7)$$

According to the first-order optimality condition, the gradient of the objective function of (7) with respect to each singular value should vanish. For  $\log \det$  function, we have

$$\frac{1}{1 + \sigma_i} + \beta(\sigma_i - \sigma_{i,A}) = 0 \text{ s.t. } \sigma_i \geq 0. \quad (8)$$

The above equation is quadratic and gives two roots. If  $\sigma_{i,A} = 0$ , the minimizer  $\sigma_i^*$  will be 0; otherwise, there exists a unique minimizer. Finally, we obtain the update of  $X$  variable with

$$X^{t+1} = U \text{diag}(\sigma^*) V^T. \quad (9)$$

Then we fix the values at the observed entries and obtain

$$X^{t+1} = \mathcal{P}_{\Omega^c}(X^{t+1}) + \mathcal{P}_\Omega(M). \quad (10)$$

To update  $Y$ , we need to solve

$$\min_Y l_{\mathcal{R}_+}(Y) + \frac{\mu^t}{2} \|X^{t+1} - Y + \frac{Z^t}{\mu^t}\|_F^2, \quad (11)$$

which yields the updating rule

$$Y^{t+1} = \max(X^{t+1} + Z^t/\mu^t, 0). \quad (12)$$

Here  $\max(\cdot)$  is an element-wise operator. The complete procedure is outlined in Algorithm 1.

To use the estimated matrix  $\hat{X}$  to make recommendation for user  $i$ , we just sort  $i$ 's non-purchased/-rated items based on their scores in decreasing order and recommend the Top- $N$  items.

Table 1: The datasets used in evaluation

dataset	#users	#items	#trns	rsize	csize	density	ratings
Delicious	1300	4516	17550	13.50	3.89	0.29%	-
lastfm	8813	6038	332486	37.7	55.07	0.62%	-
BX	4186	7733	182057	43.49	23.54	0.56%	-
ML100K	943	1682	100000	106.04	59.45	6.30%	1-10
Netflix	6769	7026	116537	17.21	16.59	0.24%	1-5
Yahoo	7635	5252	212772	27.87	40.51	0.53%	1-5

The “#users”, “#items”, “#trns” columns show the number of users, number of items and number of transactions, respectively, in each dataset. The “rsize” and “csize” columns are the average number of ratings for each user and on each item (i.e., row density and column density of the user-item matrix), respectively, in each dataset. Column corresponding to “density” shows the density of each dataset (i.e., density=#trns/(#users×#items)). The “ratings” column is the rating range of each dataset with granularity 1.

## Experimental Evaluation

### Datasets

We evaluate the performance of our method on six different real datasets whose characteristics are summarized in Table 1. These datasets are from different sources and at different sparsity levels. They can be broadly categorized into two classes.

The first class includes Delicious, lastfm and BX. These three datasets have only implicit feedback (e.g., listening history), i.e., they are represented by binary matrices. In particular, Delicious was derived from the bookmarking and tagging information from a set of  $2K$  users from Delicious social bookmarking system<sup>1</sup> such that each URL was bookmarked by at least 3 users. Lastfm corresponds to music artist listening information which was obtained from the last.fm online music system<sup>2</sup>, in which each music artist was listened to by at least 10 users and each user listened to at least 5 artists. BX is a subset from the Book-Crossing dataset<sup>3</sup> such that only implicit interactions were contained and each book was read by at least 10 users.

The second class contains ML100K, Netflix and Yahoo. All these datasets contain multi-value ratings. Specifically, the ML100K dataset corresponds to movie ratings and is a subset of the MovieLens research project<sup>4</sup>. The Netflix is a subset of data extracted from Netflix Prize dataset<sup>5</sup> and each user rated at least 10 movies. The Yahoo dataset is a subset obtained from Yahoo!Movies user ratings<sup>6</sup>. In this dataset, each user rated at least 5 movies and each movie was rated by at least 3 users.

### Evaluation Methodology

We employ 5-fold Cross-Validation to demonstrate the efficacy of our proposed approach. For each run, each of the

<sup>1</sup><http://www.delicious.com>

<sup>2</sup><http://www.last.fm>

<sup>3</sup><http://www.informatik.uni-freiburg.de/cziegler/BX/>

<sup>4</sup><http://grouplens.org/datasets/movielens/>

<sup>5</sup><http://www.netflixprize.com/>

<sup>6</sup><http://webscope.sandbox.yahoo.com/catalog.php?datatype=r>

Table 2: Comparison of Top-N recommendation algorithms

method	Delicious						lastfm					
	params			HR	ARHR	params			HR	ARHR		
ItemKNN	300	-	-	-	0.300	0.179	100	-	-	-	0.125	0.075
PureSVD	1000	10	-	-	0.285	0.172	200	10	-	-	0.134	0.078
WRMF	250	5	-	-	0.330	0.198	100	3	-	-	0.138	0.078
BPRKNN	1e-4	0.01	-	-	0.326	0.187	1e-4	0.01	-	-	0.145	0.083
BPRMF	300	0.1	-	-	0.335	0.183	100	0.1	-	-	0.129	0.073
SLIM	10	1	-	-	0.343	0.213	5	0.5	-	-	0.141	0.082
LorSLIM	10	1	3	3	0.360	0.227	5	1	3	3	0.187	0.105
Our	250	4	-	-	<b>0.452</b>	<b>0.281</b>	0.03	1.5	-	-	<b>0.467</b>	<b>0.236</b>
method	BX						ML100K					
	params			HR	ARHR	params			HR	ARHR		
ItemKNN	400	-	-	-	0.045	0.026	10	-	-	-	0.287	0.124
PureSVD	3000	10	-	-	0.043	0.023	100	10	-	-	0.324	0.132
WRMF	400	5	-	-	0.047	0.027	50	1	-	-	0.327	0.133
BPRKNN	1e-3	0.01	-	-	0.047	0.028	2e-4	1e-4	-	-	0.359	0.150
BPRMF	400	0.1	-	-	0.048	0.027	200	0.1	-	-	0.330	0.135
SLIM	20	0.5	-	-	0.050	0.029	2	2	-	-	0.343	0.147
LorSLIM	50	0.5	2	3	0.052	0.031	10	8	5	3	0.397	0.207
Our	1.2e-3	1.3	-	-	<b>0.132</b>	<b>0.059</b>	6e-3	2.5	-	-	<b>0.578</b>	<b>0.265</b>
method	Netflix						Yahoo					
	params			HR	ARHR	params			HR	ARHR		
ItemKNN	200	-	-	-	0.156	0.085	300	-	-	-	0.318	0.185
PureSVD	500	10	-	-	0.158	0.089	2000	10	-	-	0.210	0.118
WRMF	300	5	-	-	0.172	0.095	100	4	-	-	0.250	0.128
BPRKNN	2e-3	0.01	-	-	0.165	0.090	0.02	1e-3	-	-	0.310	0.182
BPRMF	300	0.1	-	-	0.140	0.072	300	0.1	-	-	0.308	0.180
SLIM	5	1.0	-	-	0.173	0.098	10	1	-	-	0.320	0.187
LorSLIM	10	3	5	3	0.196	0.111	10	1	2	3	0.334	0.191
Our	0.015	1.2	-	-	<b>0.326</b>	<b>0.167</b>	5e-3	1.1	-	-	<b>0.574</b>	<b>0.268</b>

The parameters for each method are as follows: ItemKNN: the number of neighbors  $k$ ; PureSVD: the number of singular values and the number of iterations during SVD; WRMF: the dimension of the latent space and the weight on purchases; BPRKNN: the learning rate and regularization parameter  $\lambda$ ; BPRMF: the dimension of the latent space and learning rate; SLIM: the  $l_2$ -norm regularization parameter  $\beta$  and the  $l_1$ -norm regularization parameter  $\lambda$ ; LorSLIM: the  $l_2$ -norm regularization parameter  $\beta$ , the  $l_1$ -norm regularization parameter  $\lambda$ , the nuclear norm regularization parameter  $z$  and the auxiliary parameter  $\rho$ . Our: auxiliary parameters  $\mu^0$  and  $\gamma$ .  $N$  in this table is 10. Bold numbers are the best performance in terms of HR and ARHR for each dataset.

datasets is split into training and test sets by randomly selecting one of the non-zero entries for each user to be part of the test set<sup>7</sup>. The training set is used to train a model, then a size- $N$  ranked list of recommended items for each user is generated. The evaluation of the model is conducted by comparing the recommendation list of each user and the item of that user in the test set. For the following results reported in this paper,  $N$  is equal to 10.

Top- $N$  recommendation is more like a ranking problem rather than a prediction task. The recommendation quality is measured by the hit-rate (HR) and the average reciprocal hit-rank (ARHR) (Deshpande and Karypis 2004). They directly measure the performance of the model on the ground truth data, i.e., what users have already provided feedback for. As pointed out in (Ning and Karypis 2011), they are the most direct and meaningful measures in Top- $N$  recommendation

scenarios. HR is defined as

$$HR = \frac{\#\text{hits}}{\#\text{users}}, \quad (13)$$

where #hits is the number of users whose item in the test set is recommended (i.e., hit) in the size- $N$  recommendation list, and #users is the total number of users. An HR value of 1.0 indicates that the algorithm is able to always recommend the hidden item, whereas an HR value of 0.0 denotes that the algorithm is not able to recommend any of the hidden items.

A drawback of HR is that it treats all hits equally regardless of where they appear in the Top- $N$  list. ARHR addresses it by rewarding each hit based on where it occurs in the Top- $N$  list, which is defined as follows:

$$ARHR = \frac{1}{\#\text{users}} \sum_{i=1}^{\#\text{hits}} \frac{1}{p_i}, \quad (14)$$

where  $p_i$  is the position of the test item in the ranked Top- $N$  list for the  $i$ -th hit. That is, hits that occur earlier in

<sup>7</sup>We use the same data as in (Cheng, Yin, and Yu 2014), with partitioned datasets kindly provided by Yao Cheng.

the ranked list are weighted higher than those occur later, and thus ARHR measures how strongly an item is recommended. The highest value of ARHR is equal to the hit-rate and occurs when all the hits occur in the first position, whereas the lowest value is equal to HR/N when all the hits occur in the last position of the list.

## Comparison Algorithms

We compare the performance of the proposed method<sup>8</sup> with seven other state-of-the-art Top-N recommendation algorithms, including the item neighborhood-based collaborative filtering method ItemKNN (Deshpande and Karypis 2004), two MF-based methods PureSVD (Cremonesi, Koren, and Turrin 2010) and WRMF (Hu, Koren, and Volinsky 2008), two ranking/retrieval criteria based methods BPRMF and BPRKNN (Rendle et al. 2009), SLIM (Ning and Karypis 2011), and LorSLIM (Cheng, Yin, and Yu 2014).

## Experimental Results

### Top-N recommendation performance

We report the comparison results with other competing methods in Table 2. The results show that our algorithm performs better than all the other methods across all the datasets. Specifically, in terms of HR, our method outperforms ItemKNN, PureSVD, WRMF, BPRKNN, BPRMF, SLIM and LorSLIM by 134.75%, 145.36%, 125.35%, 114.22%, 127.72%, 110.55%, 85.48% on average, respectively, over all the six datasets; with respect to ARHR, ItemKNN, PureSVD, WRMF, BPRKNN, BPRMF, SLIM and LorSLIM are improved by 108.94%, 123.00%, 107.90%, 92.47%, 110.78%, 86.20%, 31.06% on average, respectively. Moreover, the HR improvements achieved by our method against the next best performing scheme (i.e., LorSLIM) are quite substantial; the enhancements are 0.092, 0.28, 0.08, 0.181, 0.13, 0.24, respectively.

Among the seven state-of-the-art algorithms, LorSLIM is substantially better than the others. Moreover, SLIM is a little better than others except on lastfm and ML100K among the rest six methods. Then BPRKNN performs best among the remaining five methods on average. Among the three MF-based models, BPRMF and WRMF have similar performance on most datasets and are much better than PureSVD on all datasets except on lastfm and ML100K.

### Recommendation for Different Top-N

Figure 1 shows the performance achieved by the various methods for different values of  $N$  for all six datasets. It demonstrates that the proposed method outperforms other algorithms in all scenarios. What is more, it is evident that the difference in performance between our approach and the other methods are consistently significant. It is interesting to note that LorSLIM, the second most competitive method, may be worse than some of the rest methods when  $N$  is large.

<sup>8</sup>The implementation of our method is available at: [https://github.com/sckangz/recom\\_mc](https://github.com/sckangz/recom_mc).

## Matrix Reconstruction

We compare our method with LorSLIM by looking at how they reconstruct the user-item matrix. We take ML100K as an example, whose density is 6.30% and the mean for those non-zero elements is 3.53. The reconstructed matrix from LorSLIM  $\hat{X} = XW$  has a density of 13.61%, whose non-zero values have a mean of 0.046. For those 6.30% non-zero entries in  $X$ ,  $\hat{X}$  recovers 70.68% of them and their mean value is 0.0665. This demonstrates that lots of information is lost. On the contrary, our approach fully preserves the original information thanks to the constraint condition in our model. Our method recovers all zero values with a mean of 0.554, which is much higher than 0.046. This suggests that our method recovers  $X$  better than LorSLIM. This may explain the superior performance of our method.

## Parameter Tuning

Although our model is parameter-free, we introduce the auxiliary parameter  $\mu$  during the optimization. In alternating direction method of multipliers (ADMM) (Yang and Yuan 2013),  $\mu$  is fixed and it is not easy to choose an optimal value to balance the computational cost. Thus, a dynamical  $\mu$ , increasing at a rate of  $\gamma$ , is preferred in real applications.  $\gamma > 1$  controls the convergence speed. The larger  $\gamma$  is, the fewer iterations are to obtain the convergence, but meanwhile we may lose some precision. We show the effects of different initializations  $\mu^0$  and  $\gamma$  on HR on dataset Delicious in Figure 2. It illustrates that our experimental results are not sensitive to them, which is reasonable since they are auxiliary parameters controlling mainly the convergence speed. In contrast, LorSLIM needs to tune four parameters, which are time consuming and not easy to operate.

## Efficiency Analysis

The time complexity of our algorithm is mainly from SVD. Exact SVD of a  $m \times n$  matrix has a time complexity of  $O(\min\{mn^2, m^2n\})$ , in this paper we seek a low-rank matrix and thus only need a few principal singular vectors/values. Packages like PROPACK (Larsen 2004) can compute a rank  $k$  SVD with a cost of  $O(\min\{m^2k, n^2k\})$ , which can be advantageous when  $k \ll m, n$ . In fact, our algorithm is much faster than LorSLIM. Among the six datasets, ML100K and lastfm datasets have the smallest and largest sizes, respectively. Our method needs 9s and 5080s, respectively, on these two datasets, while LorSLIM takes 617s and 32974s. The time is measured on the same machine with an Intel Xeon E3-1240 3.40GHz CPU that has 4 cores and 8GB memory, running Ubuntu and Matlab (R2014a). Furthermore, without losing too much accuracy,  $\gamma$  can speed up our algorithm considerably. This is verified by Figure 3, which shows the computational time of our method on Delicious with varying  $\gamma$ .

## Conclusion

In this paper, we present a matrix completion based method for the Top-N recommendation problem. The proposed method recovers the user-item matrix by solving a rank minimization problem. To better approximate the rank, a non-

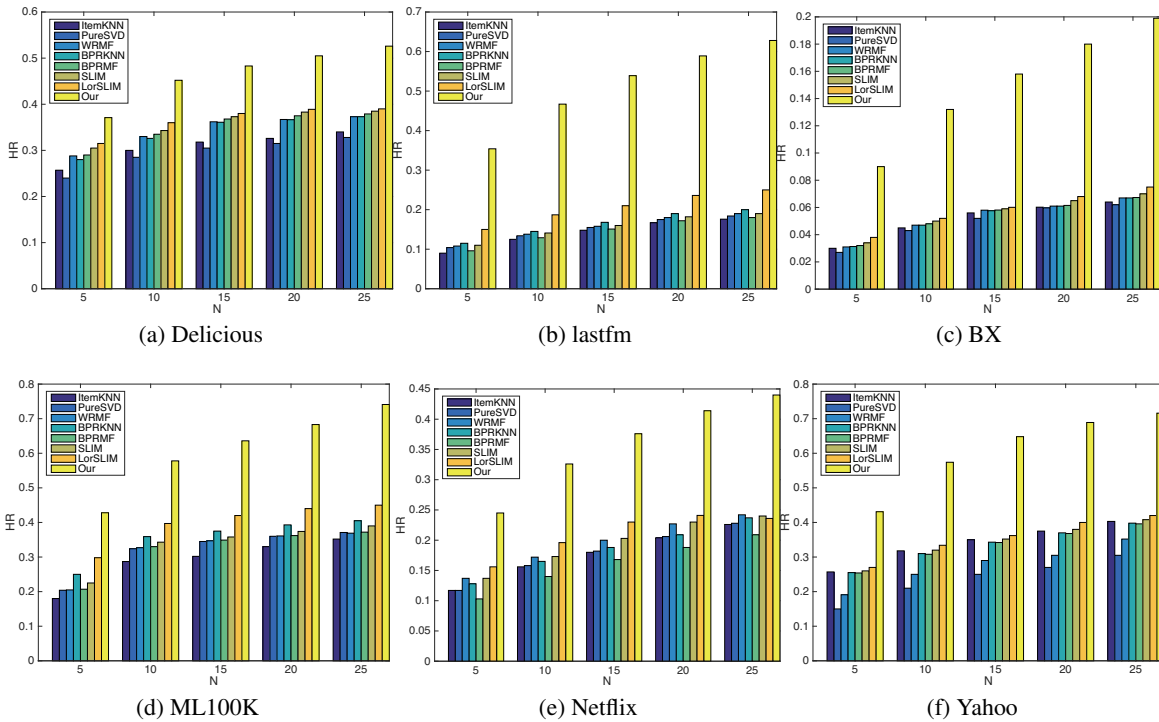


Figure 1: Performance for Different Values of  $N$ .

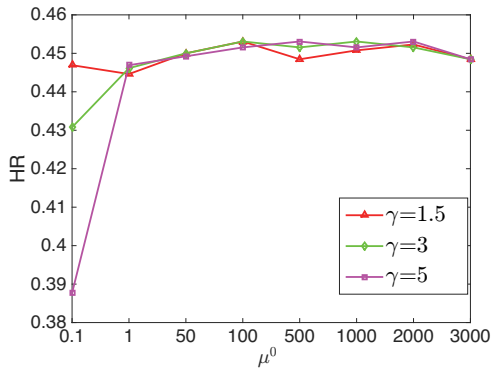


Figure 2: Influence of  $\mu^0$  and  $\gamma$  on HR for Delicious dataset.

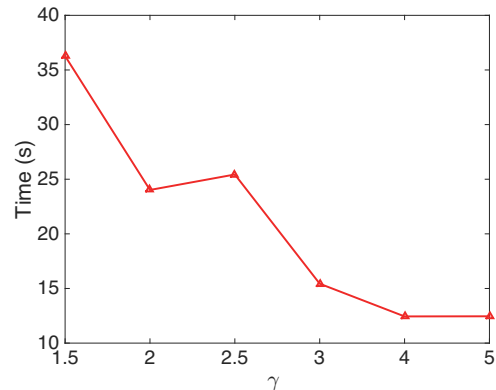


Figure 3: Influence of  $\gamma$  on time.

convex function is applied. We conduct a comprehensive set of experiments on multiple datasets and compare its performance against that of other state-of-the-art Top- $N$  recommendation algorithms. It turns out that our algorithm generates high quality recommendations, which improves the performance of the rest of methods considerably. This makes our approach usable in real-world scenarios.

### Acknowledgements

This work is supported by US National Science Foundation Grants IIS 1218712. Q. Cheng is the corresponding author.

### References

- Breese, J. S.; Heckerman, D.; and Kadie, C. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, 43–52. Morgan Kaufmann Publishers Inc.
- Candès, E. J., and Recht, B. 2009. Exact matrix completion via convex optimization. *Foundations of Computational mathematics* 9(6):717–772.
- Cheng, Y.; Yin, L.; and Yu, Y. 2014. Lorslim: Low rank sparse linear methods for top- $n$  recommendations. In *Data*

- Mining (ICDM), 2014 IEEE International Conference on*, 90–99. IEEE.
- Cremonesi, P.; Koren, Y.; and Turrin, R. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, 39–46. ACM.
- Degemmis, M.; Lops, P.; and Semeraro, G. 2007. A content-collaborative recommender that exploits wordnet-based user profiles for neighborhood formation. *User Modeling and User-Adapted Interaction* 17(3):217–255.
- Deshpande, M., and Karypis, G. 2004. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)* 22(1):143–177.
- Desrosiers, C., and Karypis, G. 2011. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*. Springer. 107–144.
- Fazel, M.; Hindi, H.; and Boyd, S. P. 2003. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *American Control Conference, 2003. Proceedings of the 2003*, volume 3, 2156–2162. IEEE.
- Good, N.; Schafer, J. B.; Konstan, J. A.; Borchers, A.; Sarwar, B.; Herlocker, J.; and Riedl, J. 1999. Combining collaborative filtering with personal agents for better recommendations. In *AAAI/IAAI*, 439–446.
- Hu, Y.; Koren, Y.; and Volinsky, C. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, 263–272. IEEE.
- Kang, Zhao ang Peng, C., and Cheng, Q. 2015. Robust pca via nonconvex rank approximation. In *Data Mining (ICDM), 2015 IEEE International Conference on*, 211–220. IEEE.
- Kang, Z.; Peng, C.; Cheng, J.; and Cheng, Q. 2015. Logdet rank minimization with application to subspace clustering. *Computational intelligence and neuroscience* 2015:68.
- Kang, Z.; Peng, C.; and Cheng, Q. 2015a. Robust subspace clustering via smoothed rank approximation. *SIGNAL PROCESSING LETTERS, IEEE* 22(11):2088–2092.
- Kang, Z.; Peng, C.; and Cheng, Q. 2015b. Robust subspace clustering via tighter rank approximation. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM'15)*, 393–401. ACM.
- Larsen, R. M. 2004. Propack-software for large and sparse svd calculations. Available online. URL <http://sun.stanford.edu/rmunk/PROPACK> 2008–2009.
- Lee, J.; Bengio, S.; Kim, S.; Lebanon, G.; and Singer, Y. 2014. Local collaborative ranking. In *Proceedings of the 23rd international conference on World wide web*, 85–96. ACM.
- Li, J., and Zaïane, O. R. 2004. Combining usage, content, and structure data to improve web site recommendation. In *E-Commerce and Web Technologies*. Springer. 305–315.
- Linden, G.; Smith, B.; and York, J. 2003. Amazon.com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE* 7(1):76–80.
- Lu, C.; Tang, J.; Yan, S.; and Lin, Z. 2014. Generalized nonconvex nonsmooth low-rank minimization. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, 4130–4137. IEEE.
- Melville, P.; Mooney, R. J.; and Nagarajan, R. 2002. Content-boosted collaborative filtering for improved recommendations. In *AAAI/IAAI*, 187–192.
- Ning, X., and Karypis, G. 2011. Slim: Sparse linear methods for top-n recommender systems. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, 497–506. IEEE.
- Pan, R.; Zhou, Y.; Cao, B.; Liu, N. N.; Lukose, R.; Scholz, M.; and Yang, Q. 2008. One-class collaborative filtering. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, 502–511. IEEE.
- Recht, B.; Xu, W.; and Hassibi, B. 2008. Necessary and sufficient conditions for success of the nuclear norm heuristic for rank minimization. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, 3065–3070. IEEE.
- Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 452–461. AUAI Press.
- Ricci, F.; Rokach, L.; and Shapira, B. 2011. *Introduction to recommender systems handbook*. Springer.
- Shi, X., and Yu, P. S. 2011. Limitations of matrix completion via trace norm minimization. *ACM SIGKDD Explorations Newsletter* 12(2):16–20.
- Srebro, N., and Salakhutdinov, R. R. 2010. Collaborative filtering in a non-uniform world: Learning with the weighted trace norm. In *Advances in Neural Information Processing Systems*, 2056–2064.
- Yang, J., and Yuan, X. 2013. Linearized augmented lagrangian and alternating direction methods for nuclear norm minimization. *Mathematics of Computation* 82(281):301–329.
- Yu, K.; Zhu, S.; Lafferty, J.; and Gong, Y. 2009. Fast non-parametric matrix factorization for large-scale collaborative filtering. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, 211–218. ACM.
- Zhong, X.; Xu, L.; Li, Y.; Liu, Z.; and Chen, E. 2015. A non-convex relaxation approach for rank minimization problems. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.