# Improving Top-N Recommendation with Heterogeneous Loss

**Feipeng Zhao** and **Yuhong Guo**
Department of Computer and Information Sciences
Temple University, Philadelphia, PA 19122, USA
{feipeng.zhao, yuhong}@temple.edu

## Abstract

Personalized top-N recommendation systems have great impact on many real world applications such as E-commerce platforms and social networks. Most existing methods produce personalized top-N recommendations by minimizing a specific uniform loss such as pairwise ranking loss or pointwise recovery loss. In this paper, we propose a novel personalized top-N recommendation approach that minimizes a combined heterogeneous loss based on linear self-recovery models. The heterogeneous loss integrates the strengths of both pairwise ranking loss and pointwise recovery loss to provide more informative recommendation predictions. We formulate the learning problem with heterogeneous loss as a constrained convex minimization problem and develop a projected stochastic gradient descent optimization algorithm to solve it. We evaluate the proposed approach on a set of personalized top-N recommendation tasks. The experimental results show the proposed approach outperforms a number of state-of-the-art methods on top-N recommendation.

## 1 Introduction

In the era of Internet, online shopping has become a close part of people's daily life. To facilitate consumers' online shopping experience, help consumers to find their interested items from the huge amount of online items and hence encourage online item purchases, effective top-N commercial recommendation systems, which suggest a small list of items that best match each consumer's taste from the large amount of product items, have become increasingly important. Top-N recommendation systems automatically predict the recommendation scores over each item in the whole product item pool for each consumer, and recommend the items with high scores to the consumers. They can be widely used in many real world applications. For example, Amazon.com has a huge amount of online products, while consumers may not know all the special items they would like. A top-N recommendation system, which can effectively suggest a set of personalized selected products for each consumer, can significantly increase the user-item purchase probability. Similarly, Netflix Inc. has thousands of online movies and TV shows, a personalized recommendation for users can help them to find the videos they may like. Yahoo! and Apple Itunes have millions of musics, Top-N recommendation systems can help users to find the musics they may like and provide a recommendation list to each user. In summary, an effective top-N personalized recommendation system can have significant real world commercial impacts.

Many methods have been developed in the literature to build top-N recommendation systems [Ricci *et al.*, 2011]. One classical technique is collaborative filtering (CF) [Schafer *et al.*, 2007; Su and Khoshgoftaar, 2009], which models the relationships between users and the correlations between items to identify new user-items relationship scores. Two main types of collaborative filtering methods include neighborhood-based CF [Sarwar *et al.*, 2000; 2001; Deshpande and Karypis, 2004; Verstrepen and Goethals, 2014] and model-based CF [Rennie and Srebro, 2005; Rendle *et al.*, 2009; Cremonesi *et al.*, 2010; Sindhwani *et al.*, 2010; Hu *et al.*, 2008; Shi *et al.*, 2012; Liu and Aberer, 2014]. Standard model-based CF methods perform matrix factorization to complete the missing recommendation entries, which exploits the low-dimensional subspace representations of the users and items [Hu *et al.*, 2008; Weimer *et al.*, 2008; Yun *et al.*, 2014b]. Besides CF, sparse aggregation methods that exploit the linear correlations between items have also been explored in a few works [Ning and Karypis, 2011; Cheng *et al.*, 2014; Kabbur *et al.*, 2013; Christakopoulou and Karypis, 2014] to improve top-N recommendation performance. These methods have mainly pursued recommendation score recovery by minimizing the *pointwise comparison loss* between the reconstructed user-item matrix and the observed incomplete user-item matrix.

Recently, pairwise ranking methods that directly capture users' pairwise preference structures on product items have demonstrated good performance for top-N recommendation [Weimer *et al.*, 2008; Steck, 2010; Chen and Pan, 2013; Aiolli, 2014; Park *et al.*, 2015]. These methods have explored different ranking losses as the optimization targets, including the normalized discounted cumulative gain [Weimer *et al.*, 2008], the AUC scores [Aiolli, 2014] and the max-margin loss [Park *et al.*, 2015]. Nevertheless, different types of losses have different strengths in producing the missing recommendation scores. The pointwise losses enforce the entrywise consistency between the reconstructed recommendation matrix and the original matrix, while the pairwise ranking losses enforce the preference structure consistency. Majority of the existing methods however have focused on using a uniform type of recovery loss, which limits their abilities of integrating the complementary strengths of different types of losses.

In this paper, we propose a novel personalized top-N recommendation approach that uses a combined heterogeneous loss function based on linear self-recovery models. The heterogeneous loss integrates the strengths of pairwise ranking loss and pointwise recovery loss to enforce both entrywise consistency and preference structure consistency between the recommendation recovery and the observations, which leads to improved recommendation predictions. We formulate the learning problem with this heterogeneous loss as a constrained convex minimization problem and develop a projected stochastic gradient descent optimization algorithm to solve it. The proposed approach is evaluated on a set of personalized top-N recommendation tasks. The experimental results show the proposed approach outperforms a number of state-of-the-art methods on top-N recommendation.

## 2 Related Works

Existing top-N recommendation methods have mostly focused on minimizing a uniform type of recommendation recovery loss to predict the recommendation scores. Based on the type of the losses, these recommendation systems can be categories into two main groups. The first group exploits pointwise comparison losses and the second group exploits pairwise ranking losses.

**Pointwise comparison methods.** Many traditional top-N recommendation algorithms perform learning by minimizing the pointwise (entrywise) divergence of the reconstructed recommendation matrix from the original observation matrix. These include the collaborative filtering methods [Schafer *et al.*, 2007; Su and Khoshgoftaar, 2009; Hu *et al.*, 2008; Weimer *et al.*, 2008; Yun *et al.*, 2014b], and the sparse linear aggregation methods [Ning and Karypis, 2011; Cheng *et al.*, 2014; Kabbur *et al.*, 2013; Christakopoulou and Karypis, 2014]. For example, a weighted regularized matrix factorization (WRMF) model has been developed in [Hu *et al.*, 2008], which minimizes the pointwise difference between the reconstructed matrix produced by the inner product of the user and item latent representations and the training matrix. Similarly, the work in [Yun *et al.*, 2014b] enforces the pointwise similarity between the training matrix and reconstructed matrix. It uses a non-locking parallel computing algorithm to perform matrix completion. In [Ning and Karypis, 2011], a sparse linear method (SLIM) was proposed to recover the missing recommendation entries. It exploits the item-item correlations to reconstruct the incomplete recommendation matrix and minimizes the pointwise reconstruction loss between the reconstructed matrix and the input training matrix.

**Pairwise ranking methods.** Given a pair of items, a user naturally will prefer one to another, which forms the important pairwise preference structure of the recommendation data. The pointwise comparison methods however ignored this user personalized preference information. Recently, pairwise ranking methods are proposed to produce top-N recommendation systems by optimizing the preference structure consistency between the original matrix and the reconstructed recommendation matrix [Weimer *et al.*, 2008; Steck, 2010; Chen and Pan, 2013; Aiolli, 2014; Park *et al.*, 2015; Rendle *et al.*, 2009]. Pairwise ranking methods treat training data as a set of triplet instances; for example, the triplet $(i, j, k)$ is an instance that encodes the $i$-th user's preference to item $j$ over item $k$. Different pairwise ranking losses have been exploited in these works. For example, the pairwise ranking methods in [Rendle *et al.*, 2009; Aiolli, 2014] optimize AUC scores; the work in [Weimer *et al.*, 2008] optimizes a normalized discounted cumulative gain; Yun *et al.* (2014a) explored the connection between the metric discounted cumulative gain and the binary classification to change the ranking problem into binary classification problems; Park *et al.* (2015) proposed a large-scale collaborative ranking method that exploits a max-margin hinge ranking loss to minimize the ranking risk in the reconstructed recommendation matrix. Nevertheless, these pairwise ranking methods ignored the entrywise consistency between the reconstructed matrix and the original matrix.

Different from all these existing methods, our proposed approach will perform top-N recommendations by optimizing a heterogeneous loss that integrates the strengths of both a pointwise comparison loss and a pairwise ranking loss.

## 3 Approach

In this section, we present a novel personalized top-N recommendation approach that minimizes a combined heterogeneous loss within a general learning framework. We assume a partially observed user-item recommendation/purchase matrix $X \in \mathbb{R}^{n\times m}$ over $n$ users and $m$ items is given. For implicit feedbacks, the recommendation/rating values are within $\{0, 1\}$, where the entry $X_{ij} = 1$ indicates a transaction or purchase record for the $i$-th user on the $j$-th item. The entry $X_{ij} = 0$ on the other hand means either the $i$-th user never purchased the $j$-th item or the transaction record has been removed and need to be predicted. We aim to identify the most interesting items for each user from his unrecommended/unpurchased list of items. Below we first introduce the general learning framework for top-N recommendations and then instantiate it with novel objective losses. We finally present a stochastic gradient descent algorithm to solve the recommendation problem formulated.

### 3.1 A General Learning Framework for Top-N Recommendation

Given the input user-item recommendation/purchase matrix $X$, a general framework for top-N recommendation performs learning by minimizing a regularized reconstruction loss function:

$$\min_{W} \ \mathcal{L}(X, \widehat{X}(W)) + R(W) \tag{1}$$

where $\widehat{X}$ denotes the reconstructed recommendation matrix with a parametric model with parameter matrix $W$, $\mathcal{L}(\cdot)$ denotes a convex reconstruction loss function and $R(\cdot)$ denotes a regularization function. By substituting $\mathcal{L}(\cdot)$ with different loss functions and employing different reconstruction models (denoted by the parameter $W$), many existing methods can be produced from this general framework as specific examples. For example, by using a matrix factorization model that reconstructs $X$ as $\widehat{X} = UV^\top$ and a least squares pointwise loss function $\mathcal{L}(X, \widehat{X}) = \|X - \widehat{X}\|_F^2$, we can produce the simple pure singular value decomposition based (PureSVD) matrix factorization method [Cremonesi *et al.*, 2010] with proper constraints. Similarly, by reconstructing $X$ with a linear aggregation model $\widehat{X} = XW^\top$ and using a least squares pointwise loss function $\mathcal{L}(\cdot)$ and an integrated $\ell_2$ and $\ell_1$ norm regularization function $R(W) = \beta\|W\|_F^2 + \lambda\|W\|_1$, the sparse linear method (SLIM) [Ning and Karypis, 2011] can be produced from the framework with additional proper constraints.

Moreover, the state-of-the-art pairwise ranking methods can also be produced as specific examples from this framework. For example, given the set of preference triplets $\Omega$, by using a matrix factorization model to reconstruct $X$ as $\widehat{X} = UV^\top$, a max-margin hinge loss function $\mathcal{L}_{rank}(x) = \max(0, 1 - x)$, and $\ell_2$ norm regularizers on $U$ and $V$, we can produce the large scale pairwise ranking method (AltSVM) in [Park *et al.*, 2015] as below:

$$\min_{U,V} \sum_{(i,j,k)\in\Omega} \mathcal{L}_{rank}\left(Y_{ijk}(X) \cdot U_i(V_j - V_k)^\top\right) + \frac{\beta}{2}(\|U\|_F^2 + \|V\|_F^2) \qquad (2)$$

where $Y_{ijk}$ is a function of $X$ such that $Y_{ijk} = 1$ if user $i$ prefers item $j$ over item $k$ in $X$ and $Y_{ijk} = -1$ otherwise; $\|\cdot\|_F$ denotes the matrix Frobenius norm; $U_i$ denotes the $i$-th row of $U$ and $V_j$ denotes the $j$-th row of $V$.

Our proposed approach can be conveniently formulated within this general learning framework as well. We integrate two types of losses into a novel heterogeneous loss based on linear self-recovery models.

## 3.2 Novel Heterogeneous Loss

Pairwise ranking methods have demonstrated great top-N recommendation performance in the literature [Park *et al.*, 2015]. However, with the matrix factorization reconstruction function, the scalable learning problem produced in Eq. (2) is a non-convex optimization problem. To introduce a convenient convex formulation with pairwise ranking losses, we propose to adopt a linear self-recovery model to reconstruct $X$. In particular, we use a linear reconstruction function $\widehat{X} = XW^\top$ with constraints $W \geq 0$ and $\text{diag}(W) = 0$. This linear function exploits the statistical positive item-item correlations distributed among all the users to reconstruct the missing recommendation scores. The constraints are used to avoid trivial solutions. Then with a pairwise ranking loss function $\mathcal{L}_{rank}(\cdot)$, we can formulate the following convex pairwise ranking problem to perform top-N recommendation:

$$\min_{W} \sum_{(i,j,k)\in\Omega} \mathcal{L}_{rank}\big(Y_{ijk}(X) \cdot X_i(W_j - W_k)^\top\big) + \frac{\beta}{2}\|W\|_F^2$$
$$\text{s.t.} \quad W \geq 0, \quad \text{diag}(W) = 0 \qquad (3)$$

Note for each user $i$, if he/she purchased item $j$ but there is no purchase record for item $k$ in the training matrix $X$, we say that user $i$ prefers item $j$ over item $k$ and we have $X_{ij} > X_{ik}$. The set $\Omega$ in the formulation above contains all the triplets $(i, j, k)$ where $X_{ij} > X_{ik}$. To maintain the same pairwise preference structure, we assume that if $X_{ij} > X_{ik}$ in the original matrix $X$, one should also have $\widehat{X}_{ij} > \widehat{X}_{ik}$ in the reconstructed matrix $\hat{X}$. The loss function $\mathcal{L}_{rank}(\cdot)$ aims to encode the pairwise preference ranking divergence between the given matrix $X$ and the reconstructed matrix $\widehat{X}$. In our implementation, we use the max-margin hinge loss function as the ranking loss, i.e., $\mathcal{L}_{rank}(x) = \max(0, 1 - x)$. By minimizing such a ranking loss function, the model will increase the consistency of the pairwise preference ranking between the reconstructed matrix $\widehat{X}$ and the original matrix $X$. However, the consistency is only maintained at the pairwise relative level. It does not necessarily guarantee the good quality of pointwise reconstruction, which might consequently hamper the accurate inference of unseen pairwise relationships.

We nevertheless have an easy solution. The linear self-recovery model $\widehat{X} = XW^\top$ naturally allows one to encode the pointwise recovery loss between the entries of the reconstructed matrix $\widehat{X}$ and the original matrix $X$, e.g., $\mathcal{L}_{point}(X, \widehat{X}) = \|X - \widehat{X}\|_F^2$. Previous works [Ning and Karypis, 2011] have also demonstrated good top-N recommendation performance with a least squares pointwise recovery loss. Hence we propose to combine both the pairwise ranking loss and the pointwise recovery loss to produce a new heterogeneous loss function for top-N recommendation. Our learning problem with the heterogeneous loss function can be formulated as below:

$$\min_{W} \sum_{(i,j,k)\in\Omega} \mathcal{L}_{rank}\big(Y_{ijk}(X) \cdot X_i(W_j - W_k)^\top\big) + \frac{\alpha}{2}\|X - XW^\top\|_F^2 + \frac{\beta}{2}\|W\|_F^2 \qquad (4)$$
$$\text{s.t.} \quad W \geq 0, \quad \text{diag}(W) = 0$$

The heterogeneous loss function used in this formulation integrates two different types of losses, and is expected to capture both the pointwise similarity discrepancy and the pairwise ranking consistency discrepancy to produce a better reconstruction matrix. Moreover, the learning problem (4) remains to be a convex optimization problem.

## 3.3 Optimization Algorithm

Due to the large size of the user-item matrix, standard gradient descent algorithms are not efficient for solving the minimization problem in (4). Moreover, though the learning problem is convex, the hinge loss function for pairwise ranking is non-differentiable and subgradients are typically needed. Hence in this work, we develop a projected stochastic gradient descent (SGD) algorithm to solve our target learning problem. The SGD technique has been widely used in recommendation systems with pairwise ranking [Rendle *et al.*, 2009; Chen and Pan, 2013; Yun *et al.*, 2014b], and it can also be extended to perform parallel computations [Zinkevich *et al.*, 2011; Recht *et al.*, 2011; Yun *et al.*, 2014b].

With the pairwise ranking loss, we use the user-item-item triplet $(i, j, k)$ instances for the stochastic gradient descent procedure. In each iteration, we randomly choose a triplet $(i, j, k) \in \Omega$ and make a SGD update:

$$W_j^{\top +} \leftarrow W_j^{\top} - \eta \left\{ \begin{array}{l} \mathcal{L}'_{rank}(W_j^{\top}) + \frac{\beta}{|\Omega^j|} W_j^{\top} \\ + \frac{\alpha}{|\Omega^j|} X^{\top}(X W_j^{\top} - X_{:,j}) \end{array} \right\} \quad (5)$$

$$W_k^{\top +} \leftarrow W_k^{\top} - \eta \left\{ \begin{array}{l} \mathcal{L}'_{rank}(W_k^{\top}) + \frac{\beta}{|\Omega^k|} W_k^{\top} \\ + \frac{\alpha}{|\Omega^k|} X^{\top}(X W_k^{\top} - X_{:,k}) \end{array} \right\} \quad (6)$$

where $|\Omega^j|$ and $|\Omega^k|$ denote the number of comparisons in $\Omega$ that involve item $j$ and item $k$ respectively. $\mathcal{L}'_{rank}(W_j^{\top})$ and $\mathcal{L}'_{rank}(W_k^{\top})$ denote the derivations of $\mathcal{L}_{rank}(\cdot)$ on $W_j^{\top}$ and $W_k^{\top}$ respectively, such that

$$\mathcal{L}'_{rank}(W_j^{\top}) = Y_{ijk} \cdot 2X_i^{\top}(X_i(W_j - W_k)^{\top})$$

if $X_i(W_j - W_k)^{\top} \leq 1$ and $\mathcal{L}'_{rank}(W_j^{\top}) = 0$ otherwise. Similarly,

$$\mathcal{L}'_{rank}(W_k^{\top}) = -Y_{ijk} \cdot 2X_i^{\top}(X_i(W_j - W_k)^{\top})$$

if $X_i(W_j - W_k)^{\top} \leq 1$ and $\mathcal{L}'_{rank}(W_k^{\top}) = 0$ otherwise.

The $\eta$ parameter in the SGD update is the stepsize of gradient descent. We used a fixed small stepsize value in our experiments. The overall learning algorithm is given in Algorithm 1.

**Algorithm 1** Projected Stochastic Gradient Descent

```
Input: $\alpha > 0,\ \beta > 0,\ \eta > 0$; initialize $W^0$ as zeros.
Set $W = W^0$
for iter = 1 to MaxIter do
    1. Randomly select $(i, j, k) \in \Omega$
    2. Update $W_j$ by using (5)
    3. Project the updated $W_j$ into the feasible set:
          $W_j = \max(W_j, 0), \quad W_{jj} = 0$;
    4. Update $W_k$ by using (6)
    5. Project the updated $W_k$ into the feasible set:
          $W_k = \max(W_k, 0), \quad W_{kk} = 0$;
    6. if converge then break-out end if
end for
return $W$
```

# 4 Experimental Results

In this section, we first present the experimental setting and then report the empirical results.

## 4.1 Experimental Setup

**Datasets.** We used five datasets in our experiments: *Yahoo!* music ratings v1.0 dataset, *Yahoo!* movie ratings v1.0 dataset, *MovieLens 100k (ml-100k)* dataset, *MovieLens 1M (ml-1m)* dataset and *Netflix* dataset. *Yahoo!Music* contains ratings for songs, the ratings are supplied by users during normal interactions with Yahoo! Music services. Similarly, *Yahoo!Movie* contains ratings for different movies. In each dataset, we converted the entries with positive values to 1 and converted the user-item matrix to an implicit feedback matrix. Since the original rating matrix is sparse, for *Yahoo! Movie* and *Yahoo! Music* we kept the users with more than 20 ratings and items with more than 5 ratings. For *ml-100k* we used original implicit feedback matrix. For *ml-1m* and *Netflix* we used the same datasets as [Aiolli, 2014]. The statistic information of these datasets is reported in Table 1.

Table 1: Statistic information of the datasets: The columns of #user, #item and #transact show the numbers of users,items, and non-zero transactions respectively in each dataset. The columns of #rsize and #csize show the average number of transactions for each user and each item respectively. The column of density shows the density of non-zero transactions in each dataset.

| Dataset | #user | #item | #transact | density |
|---|---|---|---|---|
| Yahoo! Music | 2689 | 994 | 86907 | 3.95% |
| Yahoo! Movie | 2382 | 924 | 104459 | 4.75% |
| ml-100k | 943 | 1682 | 100000 | 6.30% |
| ml-1m | 3850 | 2273 | 315869 | 3.60% |
| netflix | 2979 | 2544 | 114865 | 1.50% |

**Evaluation Methods.** For each dataset, we split it into a training set and a test set. For each user, we randomly selected ten feedbacks and placed them into the test set and the rest were used as training set. After training, a ranked list of top-N items can be returned for each user according to the reconstruction scores, which were compared to the test set for performance evaluation.

We used two standard performance metrics, the top-N prediction precision (Precision@N) and the mean average precision (MAP@N), to evaluate the test performance. The top-N prediction precision for the $i$-th user is defined as

$$\text{Precision@N}(i) = \frac{1}{N} \sum_{j=1}^{N} T(i, P_i(j)) \quad (7)$$

where $T$ denotes the test matrix and $P_i$ denotes the index values of the top-N predicted items in the original matrix for the $i$-th user. The overall Precision@N value is computed as the average of Precision@N(i) over all users. MAP@N denotes the mean average precision of the top-N predictions. The average precision (AP) of the top-N prediction for the $i$-th user can be defined as

$$\text{AP@N}(i) = \frac{\sum_{j=1}^{N} \text{Precision@}j(i) \times T(i, P_i(j))}{\sum_{j=1}^{N} T(i, P_i(j))}. \quad (8)$$

MAP@N can be computed as the mean of the average precision of the top-N predictions for all users, such that MAP@N=$\frac{1}{n}\sum_i$AP@N$(i)$.

In our experiment, we randomly split each dataset into training and test matrices for five times and report the average test results in terms of these two evaluation metrics.

**Compared Methods.** We compared the proposed method with four methods developed in the literature: WRMF [Hu *et al.*, 2008], SLIM [Ning and Karypis, 2011], RobiRank [Yun *et al.*, 2014a] and AltSVM [Park *et al.*, 2015].

Table 2: Average test results of top-N recommendations for all the comparison methods. The params columns contain the parameter settings for each approach. *WRMF, RobiRank* and *AltSVM* have two parameters, the latent factor dimension and the regularization parameter. *SLIM* has two parameters, $\ell_2$ and $\ell_1$ norm regularization parameters. The proposed approach has two parameters, the weight control parameter and the regularization parameter. Bold-font indicates the best results.

| method | | | | **Yahoo! Music** | | |
|---|---|---|---|---|---|---|
| | params | | Precision@5 | Precision@10 | MAP@5 | MAP@10 |
| WRMF | 200 | 100 | $0.282 \pm 0.002$ | $0.226 \pm 0.001$ | $0.521 \pm 0.002$ | $0.482 \pm 0.002$ |
| SLIM | 100 | 1 | $0.273 \pm 0.002$ | $0.213 \pm 0.001$ | $0.504 \pm 0.003$ | $0.472 \pm 0.002$ |
| RobiRank | 10 | 100 | $0.261 \pm 0.001$ | $0.209 \pm 0.001$ | $0.461 \pm 0.002$ | $0.430 \pm 0.002$ |
| AltSVM | 500 | 1000 | $0.275 \pm 0.003$ | $0.219 \pm 0.002$ | $0.504 \pm 0.003$ | $0.470 \pm 0.002$ |
| Proposed | 5000 | 0.01 | $\mathbf{0.318 \pm 0.001}$ | $\mathbf{0.244 \pm 0.001}$ | $\mathbf{0.572 \pm 0.002}$ | $\mathbf{0.526 \pm 0.002}$ |
| method | | | | **Yahoo! Movie** | | |
| | params | | Precision@5 | Precision@10 | MAP@5 | MAP@10 |
| WRMF | 200 | 100 | $0.390 \pm 0.001$ | $0.316 \pm 0.001$ | $0.595 \pm 0.003$ | $0.554 \pm 0.003$ |
| SLIM | 100 | 1 | $0.378 \pm 0.002$ | $0.303 \pm 0.001$ | $0.577 \pm 0.003$ | $0.539 \pm 0.003$ |
| RobiRank | 10 | 100 | $0.358 \pm 0.002$ | $0.290 \pm 0.002$ | $0.520 \pm 0.003$ | $0.483 \pm 0.002$ |
| AltSVM | 500 | 1000 | $0.391 \pm 0.002$ | $0.316 \pm 0.001$ | $0.608 \pm 0.003$ | $0.561 \pm 0.003$ |
| Proposed | 5000 | 0.01 | $\mathbf{0.430 \pm 0.001}$ | $\mathbf{0.337 \pm 0.001}$ | $\mathbf{0.657 \pm 0.001}$ | $\mathbf{0.606 \pm 0.001}$ |
| method | | | | **ml-100k** | | |
| | params | | Precision@5 | Precision@10 | MAP@5 | MAP@10 |
| WRMF | 200 | 100 | $0.299 \pm 0.001$ | $0.243 \pm 0.001$ | $0.515 \pm 0.003$ | $0.481 \pm 0.002$ |
| SLIM | 200 | 1 | $0.323 \pm 0.002$ | $0.247 \pm 0.001$ | $0.553 \pm 0.003$ | $0.516 \pm 0.003$ |
| RobiRank | 20 | 100 | $0.317 \pm 0.003$ | $0.254 \pm 0.001$ | $0.520 \pm 0.003$ | $0.477 \pm 0.002$ |
| AltSVM | 200 | 1000 | $0.316 \pm 0.002$ | $0.245 \pm 0.002$ | $0.552 \pm 0.003$ | $0.516 \pm 0.003$ |
| Proposed | 5000 | 0.01 | $\mathbf{0.342 \pm 0.001}$ | $\mathbf{0.269 \pm 0.001}$ | $\mathbf{0.574 \pm 0.002}$ | $\mathbf{0.524 \pm 0.002}$ |
| method | | | | **ml-1m** | | |
| | params | | Precision@5 | Precision@10 | MAP@5 | MAP@10 |
| WRMF | 1000 | 100 | $0.193 \pm 0.002$ | $0.160 \pm 0.002$ | $0.365 \pm 0.002$ | $0.355 \pm 0.002$ |
| SLIM | 200 | 1 | $0.193 \pm 0.003$ | $0.156 \pm 0.002$ | $0.372 \pm 0.003$ | $0.359 \pm 0.002$ |
| RobiRank | 20 | 1000 | $0.188 \pm 0.002$ | $0.154 \pm 0.002$ | $0.358 \pm 0.003$ | $0.349 \pm 0.002$ |
| AltSVM | 200 | 1000 | $0.189 \pm 0.002$ | $0.159 \pm 0.002$ | $0.362 \pm 0.002$ | $0.352 \pm 0.002$ |
| Proposed | 5000 | 1 | $\mathbf{0.211 \pm 0.002}$ | $\mathbf{0.167 \pm 0.001}$ | $\mathbf{0.397 \pm 0.003}$ | $\mathbf{0.382 \pm 0.002}$ |
| method | | | | **Netflix** | | |
| | params | | Precision@5 | Precision@10 | MAP@5 | MAP@10 |
| WRMF | 200 | 100 | $0.199 \pm 0.002$ | $0.163 \pm 0.002$ | $0.374 \pm 0.002$ | $0.364 \pm 0.002$ |
| SLIM | 100 | 0.1 | $0.192 \pm 0.002$ | $0.153 \pm 0.002$ | $0.372 \pm 0.002$ | $0.363 \pm 0.002$ |
| RobiRank | 20 | 100 | $0.143 \pm 0.002$ | $0.115 \pm 0.001$ | $0.261 \pm 0.002$ | $0.267 \pm 0.001$ |
| AltSVM | 500 | 1000 | $0.195 \pm 0.003$ | $0.159 \pm 0.002$ | $0.367 \pm 0.002$ | $0.358 \pm 0.002$ |
| Proposed | 5000 | 0.1 | $\mathbf{0.220 \pm 0.002}$ | $\mathbf{0.173 \pm 0.002}$ | $\mathbf{0.416 \pm 0.003}$ | $\mathbf{0.396 \pm 0.002}$ |

**Parameters Selection.** For the proposed method, we have two parameters, $\alpha$ and $\beta$. We selected the weight control parameter $\alpha$ from $\{10, 100, 1000, 5000, 10000\}$ and selected the regularization parameter $\beta$ from $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$. For WRMF, the latent factor dimension $f$ is selected from $\{10, 20, 50, 100, 200, 500, 1000\}$ and the regularization parameter $\lambda$ is selected from $\{0.1, 1, 10, 100, 1000\}$. For SLIM, $\ell_2$ regularization parameter $\beta$ is selected from $\{1, 10, 100, 200, 500, 1000\}$ and $\ell_1$ regularization parameter $\lambda$ is selected from $\{0.001, 0.1, 1, 10\}$. For RobiRank, the latent feature dimension $f$ is selected from $\{10, 20, 50, 100, 200, 500, 1000\}$ and the regularization parameter $\lambda$ is selected from $\{1, 10, 100, 200, 500, 1000\}$. AltSVM also has two similar parameters; the latent dimension is selected from $\{100, 200, 500, 1000\}$ and the regularization parameter $\lambda$ is selected from $\{100, 1000, 10000\}$. For each approach, we report the best results and the corresponding parameter settings.

## 4.2 Experimental Results

The selected parameter settings and the experimental results in terms of Precision@N and MAP@N with $N \in \{5, 10\}$ for all the comparison approaches are reported in Table 2. We can see that among the four comparison methods, $\{$*WRMF, SLIM, RobiRank* and *AltSVM*$\}$, *RobiRank* has poor performance and it produces the most inferior results on four datasets, *Yahoo!Music, Yahoo!Movie, Netflix* and *ml-1m*. *SLIM* performs much better than *RobiRank* and it even outperforms *RobiRank* on the remaining dataset *ml-100k* across three measurements except Precision@10. The pairwise preference ranking method, *AltSVM*, produces the best results on *Yahoo!Movie* in terms of all the measurements among the four methods, and has similar performance as *SLIM* on other datasets. The *WRMF* method, which uses regularization control and applies

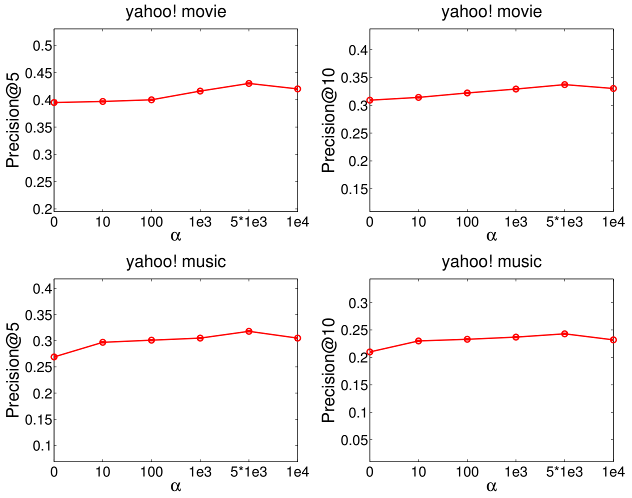

Figure 1: Parameter sensitivity analysis of $\alpha$ on Yahoo!Movie and Yahoo!Music datasets

different weights for the entries, outperforms all the other three methods on both *Yahoo!Music* and *Nexflix* in terms of the four measurements. But it has poor performance on *ml-100k*. Our proposed approach on the other hand outperforms all these four methods with remarkable margins on all the datasets across the four measurements. These results suggest the proposed approach is indeed an effective method for top-N recommendation.

## 4.3 Parameter Sensitivity Analysis

We have also conducted parameter sensitivity analysis for the proposed method on $Yahoo!Movie$ and $Yahoo!Music$ datasets. First, we tested different $\alpha$ values from $\{0, 10, 100, 1000, 5000, 10^4\}$ by setting $\beta$ to the fixed value used before. The experimental results are reported in Figure 1. We can see that for Yahoo!Movie dataset, the precision values increase when $\alpha$ increases from 0 to 5000. When $\alpha = 5000$, the model performs the best. Similar results can be observed on Yahoo!Music dataset. According to our objective function in Eq. (4), if $\alpha = 0$, the model will only use the pairwise ranking loss and will not combine the pointwise recovery loss. On the other hand, if $\alpha$ is too big, the contribution of the pairwise ranking loss can be diminished and the model will become a special version of the SLIM method. Our parameter analysis results suggest that we need a parameter in the middle to keep both types of losses, which indicates the two types of losses can complement each other. To further validate this, we have also compared our proposed approach to its variants that only use each individual loss. We call the variant that drops the pointwise recovery loss ($\alpha = 0$) as *pureRanking* and the variant that drops the pairwise ranking loss as variant of *SLIM*. The comparison results are reported in Figure 2. We can see that the proposed approach with integrated heterogeneous loss outperforms both variants that only use a uniform type of loss component. This validated our algorithm design of using heterogeneous losses.

We also conducted parameter sensitivity analysis on the $\ell_2$-norm regularization parameter $\beta$. We fixed the value of $\alpha$ as 5000 and choose $\beta$ value from $\{0, 10^{-3}, 1, 10^3, 10^6\}$. The experiment results with different $\beta$ values are reported in Figure 3. We can see that the top-N recommendation performance is not sensitive to the $\beta$ values when $\beta \leq 1000$.

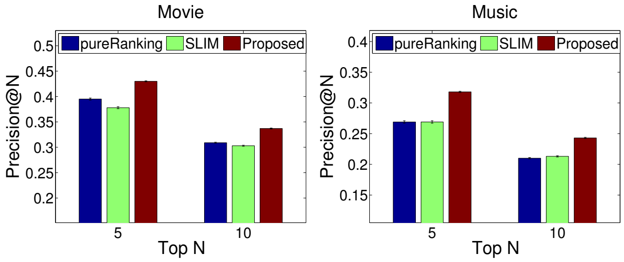

Figure 2: Comparison of proposed method and its two component variants, SLIM and pureRanking

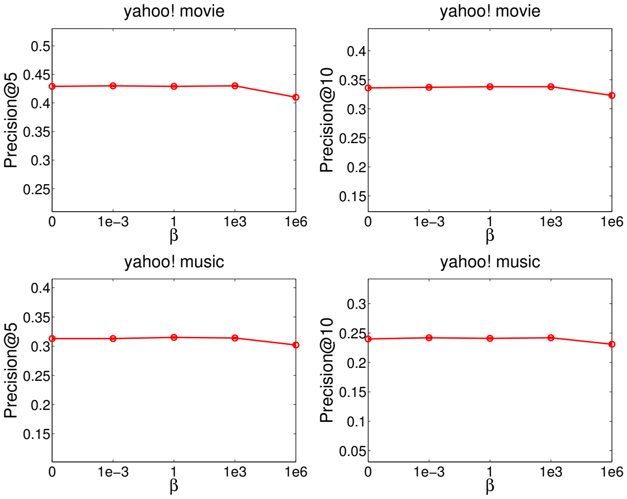

Figure 3: Parameter sensitivity analysis of $\beta$ on Yahoo!Movie and Yahoo!Music datasets

# 5 Conclusion

In this paper, we proposed a novel personalized top-N recommendation approach that exploits novel heterogeneous loss functions based on linear self-recovery models. The heterogeneous loss integrates the strengths of both pairwise ranking loss and pointwise recovery loss to enforce both entrywise consistency and pairwise preference structure consistency between the reconstructed recommendation matrix and the original observation matrix. We formulated the training problem with the heterogeneous loss as a constrained convex minimization problem and develop a projected stochastic gradient descent optimization algorithm to solve it. The proposed approach was evaluated on a set of real world personalized top-N recommendation tasks. The experimental results showed that the proposed approach not only outperforms its two variants that only used either the comparison recovery loss or the pairwise ranking loss, but also outperforms a number of state-of-the-art methods on top-N recommendation.

## References

[Aiolli, 2014] F. Aiolli. Convex AUC optimization for top-N recommendation with implicit feedback. In *Proc. of the ACM Conf. on Recommender Systems (RecSys)*, 2014.

[Chen and Pan, 2013] L. Chen and W. Pan. CoFiSet: Collaborative filtering via learning pairwise preferences over item-sets. In *Proc. of the SIAM International Conference on Data Mining (SDM)*, 2013.

[Cheng *et al.*, 2014] Y. Cheng, L. Yin, and Y. Yu. LorSLIM: Low rank sparse linear methods for top-N recommendations. In *Proc. of the IEEE International Conference on Data Mining (ICDM)*, 2014.

[Christakopoulou and Karypis, 2014] E. Christakopoulou and G. Karypis. HOSLIM: higher-order sparse linear method for top-N recommender systems. In *Proc. of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, 2014.

[Cremonesi *et al.*, 2010] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-N recommendation tasks. In *Proc. of the ACM Conference on Recommender Systems (RecSys)*, 2010.

[Deshpande and Karypis, 2004] M. Deshpande and G. Karypis. Item-based top-N recommendation algorithms. *ACM Transactions on Information Systems*, 22(1):143–177, 2004.

[Hu *et al.*, 2008] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proc. of the IEEE International Conference on Data Mining (ICDM)*, 2008.

[Kabbur *et al.*, 2013] S. Kabbur, X. Ning, and G. Karypis. FISM: Factored item similarity models for top-N recommender systems. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2013.

[Liu and Aberer, 2014] X. Liu and K. Aberer. Towards a dynamic top-N recommendation framework. In *Proc. of the ACM Conf. on Recommender Systems (RecSys)*, 2014.

[Ning and Karypis, 2011] X. Ning and G. Karypis. SLIM: sparse linear methods for top-N recommender systems. In *Proc. of the IEEE International Conference on Data Mining (ICDM)*, 2011.

[Park *et al.*, 2015] D. Park, J. Neeman, J. Xhang, and S. Sanghavi. Preference completion: Large-scale collaborative ranking from pairwise comparisons. In *Proc. of the International Conf. on Machine Learning (ICML)*, 2015.

[Recht *et al.*, 2011] B. Recht, C. Re, S. Wright, and Feng N. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

[Rendle *et al.*, 2009] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2009.

[Rennie and Srebro, 2005] J. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proc. of the International Conference on Machine Learning (ICML)*, 2005.

[Ricci *et al.*, 2011] F. Ricci, L. Rokach, and B. Shapira. Introduction to recommender systems handbook. In *Recommender Systems Handbook*, pages 1–35. Springer, 2011.

[Sarwar *et al.*, 2000] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *Proc. of the ACM Conference on Electronic Commerce*, 2000.

[Sarwar *et al.*, 2001] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proc. of the International Conference on World Wide Web (WWW)*, 2001.

[Schafer *et al.*, 2007] J. Schafer, D. Frankowski, J. Herlocker, and S. Sen. Collaborative filtering recommender systems. In *The Adaptive Web*, pages 291–324. Springer-Verlag Berlin, 2007.

[Shi *et al.*, 2012] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, and A. Hanjalic. CLiMF: Learning to maximize reciprocal rank with collaborative less-is-more filtering. In *Proc. of the ACM Conference on Recommender Systems (RecSys)*, 2012.

[Sindhwani *et al.*, 2010] V. Sindhwani, S Bucak, J. Hu, and A. Mojsilovic. One-class matrix completion with low-density factorizations. In *Proc. of the IEEE International Conference on Data Mining (ICDM)*, 2010.

[Steck, 2010] H. Steck. Training and testing of recommender systems on data missing not at random. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2010.

[Su and Khoshgoftaar, 2009] X. Su and T. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009.

[Verstrepen and Goethals, 2014] K. Verstrepen and B. Goethals. Unifying nearest neighbors collaborative filtering. In *Proc. of the ACM Conference on Recommender Systems (RecSys)*, 2014.

[Weimer *et al.*, 2008] M. Weimer, A. Karatzoglou, Q. Le, and A. Smola. COFI$^{\text{RANK}}$ - maximum margin matrix factorization for collaborative ranking. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

[Yun *et al.*, 2014a] H. Yun, P. Raman, and S. Vishwanathan. Ranking via robust binary classification. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[Yun *et al.*, 2014b] H. Yun, H. Yu, C. Hsieh, S.V.N. Vishwanathan, and I. Dhillon. NOMAD: Non-locking, stochastic multi-machine algorithm for asynchronous and decentralized matrix completion. In *Proc. of the International Conf. on Very Large Data Bases (VLDB)*, 2014.

[Zinkevich *et al.*, 2011] M. Zinkevich, M. Weimer, A. Smola, and L. Li. Parallelized stochastic gradient descent. In *Advances in Neural Information Processing Systems(NIPS)*, 2011.