One-Class Matrix Factorization: Point-Wise Regression-Based or Pair-Wise Ranking-Based?

Sheng-Wei Chen
National Taiwan University
d09944003@ntu.edu.tw

Chih-Jen Lin
National Taiwan University / MBZUAI
cjlin@csie.ntu.edu.tw

Abstract

One-class matrix factorization (MF) is an important technique for recommender systems with implicit feedback. In one widely used setting, a regression function is fit in a point-wise manner on observed and some unobserved (user, item) entries. Recently, in AAAI 2019, Chen et al. [2] proposed a pair-wise ranking-based approach for observed (user, item) entries to be compared against unobserved ones. They concluded that the pair-wise setting performs consistently better than the more traditional point-wise setting. However, after some detailed investigation, we explain by mathematical derivations that their method may perform only similar to the point-wise ones. We also identified some problems when reproducing their experimental results. After considering suitable settings, we rigorously compare point-wise and pair-wise one-class MFs, and show that the pair-wise method is actually not better. Therefore, for one-class MF, the more traditional and mature point-wise setting should still be considered. Our findings contradict the conclusions in [2] and serve as a call for caution when researchers are comparing between two machine learning methods.

CCS Concepts

• **Information systems** → **Recommender systems**; *Collaborative filtering*; *Learning to rank*; • **Computing methodologies** → *Learning from implicit feedback*.

Keywords

Reproducibility, Recommender Systems, One-Class Matrix Factorization, Point-Wise Loss, Pair-Wise Loss

ACM Reference Format:

Sheng-Wei Chen and Chih-Jen Lin. 2024. One-Class Matrix Factorization: Point-Wise Regression-Based or Pair-Wise Ranking-Based? . In *18th ACM Conference on Recommender Systems (RecSys '24), October 14–18, 2024, Bari, Italy*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3640457.3688063

1 Introduction

Recommender systems with implicit feedback are now widely deployed in many real applications. Between a user i and an item j, some (i, j) entries indicating that i likes j are observed, but the state of affairs for other entries, where i may or may not like j, is unknown. One-class collaborative filtering (OCCF) was developed to construct models for predicting unobserved (i, j) entries.

One-class matrix factorization (MF) is an important OCCF method (e.g., [5, 7, 9, 10]) and proceeds as follows. Two latent matrices are used, one for users and the other for items; the product between both matrices indicates whether a given user likes an item. With observed entries treated as positive and some unobserved entries treated as negative, the most widely used setting of one-class MF is one where the target positive/negative values are fitted to a regression function. In this work, we refer to this method as "point-wise regression-based" one-class MF.

Researchers investigating methods for constructing one-class MF models have extensively analyzed the method of selecting unobserved entries as negative data. Works such as [14] have shown that a better model is obtained if all unobserved entries are included as negative instances for training instead of only a subset. However, the huge number of unobserved (user, item) entries results in a prohibitively high training cost. Some studies (e.g., [5, 9, 14]) address this problem by showing that with some restrictions on, for example, the loss function on unobserved entries, highly efficient training algorithms can be developed.

In addition to the point-wise regression-based setting for one-class MF, another popular setting is a ranking-based one (e.g., [13]). In this approach, the aim is to have an observed (user, item) entry be ranked higher than an unobserved one. We refer to such settings as "pair-wise ranking-based" one-class MF because a pair of observed and unobserved entries are compared. Unfortunately, such approaches also have the same drawback of a prohibitive training cost if all unobserved (user, item) entries are considered.

By extending techniques for the point-wise setting, Chen et al. [2] recently addressed the computational cost problem for ranking-based one-class MF. Their experiments showed that the ranking-based setting performs consistently better than the point-wise setting. The result suggests that to achieve the state-of-the-art performance, a ranking-based one-class MF should be considered. However, for various reasons explained in this work, we are concerned about their conclusions. Through detailed investigation, we show that the ranking-based method, if not inferior, is not better than the point-wise setting. Therefore, for one-class MF, the more traditional and mature point-wise setting should still be the go-to method.

Our main contributions are summarized as follows.

- We clearly lay out the two settings (point-wise regression-based and pair-wise ranking-based), and we provide a counterargument to that of [2] about the superiority of the ranking-based setting. By some novel mathematical derivations, we show that the pair-wise ranking-based method by [2] is indeed close to a point-wise setting. Therefore, their method may only perform similarly.

RecSys '24, October 14–18, 2024, Bari, Italy

ACM ISBN 979-8-4007-0505-2/24/10
https://doi.org/10.1145/3640457.3688063

- Through the process to reproduce the experimental results in [2], we identify some possible issues in their settings. For example, they seem to report cross validation results after hyper-parameter tuning, but an independent test set should be used for performance evaluation. More importantly, the point-wise method employed in their comparison is not the most commonly used one.
- After considering suitable settings, we rigorously compare point-wise and pair-wise one-class MFs. Our results show that the point-wise setting is highly competitive.

This paper is organized as follows. Sections 2-5 correspond to the aforementioned three contributions, and Section 6 concludes this work. The programs used for our experiments are at https://www.csie.ntu.edu.tw/~cjlin/papers/ocmf_pointwise_pairwise, and the main paper with supplementary materials in the end is available at the same page.

2 MF Models for OCCF Problems

For an OCCF problem of m users and n items, the data set includes the information $r_{ij} \in \{0, 1\}$ between the user i and the item j, for $i \in \{1, \dots, m\}$ and $j \in \{1, \dots, n\}$. If $r_{ij} = 1$, we say that r_{ij} is a positive observed sample. Otherwise, $r_{ij} = 0$ is an unobserved or negative sample. In this section, we discuss various one-class MF models for OCCF. Attention was given to the ranking-based model in [2].

2.1 Regression-Based MF

Matrix factorization (MF) is an important method for OCCF model construction. Two latent matrices are considered.

$$W = \begin{bmatrix} \boldsymbol{w}_1^T \\ \vdots \\ \boldsymbol{w}_m^T \end{bmatrix} \in \mathbb{R}^{m \times k}, H = \begin{bmatrix} \boldsymbol{h}_1^T \\ \vdots \\ \boldsymbol{h}_n^T \end{bmatrix} \in \mathbb{R}^{n \times k},$$

where a small value

$$k \ll \min(m, n)$$

is the latent dimension. The goal is to find some W and H that can accurately capture whether user i likes or dislikes item j:

$$\boldsymbol{w}_i^T \boldsymbol{h}_j \approx \begin{cases} 1, & \text{if user } i \text{ likes item } j, \\ 0, & \text{otherwise.} \end{cases}$$

To obtain W and H, we typically solve an optimization problem

$$\min_{W,H} \ \text{Loss}(W, H) + \text{Reg}(W, H), \tag{1}$$

where $\text{Loss}(W, H)$ indicates the training loss and $\text{Reg}(W, H)$ is the regularization term. Now we have only the following set of observed positive entries

$$\Omega^+ = \{(i, j) \mid r_{ij} = 1\}.$$

Training is difficult in one-class MF because in the absence of negative information, a loss such as the following squared loss

$$\sum_{(i,j) \in \Omega^+} (1 - \boldsymbol{w}_i^T \boldsymbol{h}_j)^2$$

fits observed positive entries and leads to a model that predicts every (i, j) entry as positive. A popular solution (e.g., [5, 7, 9, 10, 12]) involves considering some entries in a set Ω^- as negative and uses the following loss function.

$$\sum_{(i,j) \in \Omega^+} (1 - \boldsymbol{w}_i^T \boldsymbol{h}_j)^2 + \alpha \sum_{(i,j) \in \Omega^-} (\gamma_0 - \boldsymbol{w}_i^T \boldsymbol{h}_j)^2, \tag{2}$$

where γ_0 is a value close to or equal to zero. This setting is based on the assumption that among the so many items, a user may like only a small subset. The weight coefficient α is usually smaller than one because a loss on each entry in Ω^- is considered less important than that in Ω^+.[1] Early works ([10, 12]) on one-class MF randomly selected some entries not in Ω^+ to be in the set Ω^-. However, subsequent works (e.g., [14]) have shown that considering all (user, item) entries

$$\Omega^- = \{(i, j) \mid (i, j) \notin \Omega^+\}.$$

often leads to a better model. However, this setting is expensive because of a huge number of entries in Ω^-. In particular,

$$|\Omega^-| \approx mn.$$

Thus, calculating the last term in (2) requires $O(mnk)$ operations. Several studies (e.g., [5, 9, 14]) have shown that the application of special loss functions, such as squared functions, on Ω^- can yield considerable computational savings. We provide a simple illustration by checking the following operation needed in (2).

$$\begin{aligned} \sum_{(i,j) \notin \Omega^+} (\boldsymbol{w}_i^T \boldsymbol{h}_j)^2 &= - \sum_{(i,j) \in \Omega^+} (\boldsymbol{w}_i^T \boldsymbol{h}_j)^2 + \sum_{i=1}^{m} \sum_{j=1}^{n} \boldsymbol{w}_i^T \left(\boldsymbol{h}_j \boldsymbol{h}_j^T \right) \boldsymbol{w}_i \\ &= - \sum_{(i,j) \in \Omega^+} (\boldsymbol{w}_i^T \boldsymbol{h}_j)^2 + \sum_{i=1}^{m} \boldsymbol{w}_i^T \sum_{j=1}^{n} \left(\boldsymbol{h}_j \boldsymbol{h}_j^T \right) \boldsymbol{w}_i. \end{aligned} \tag{3}$$

In the third term of (3), because k is small,

$$\sum_{j=1}^{n} \boldsymbol{h}_j \boldsymbol{h}_j^T \in \mathbb{R}^{k \times k}$$

can be computed at a computational cost of $O(nk^2)$ and stored at a memory cost of $O(k^2)$. Subsequently, the summation over i can be conducted in $O(mk^2)$. The two independent summations resolve the problem of having $O(mn)$ operations over all (i, j) entries.

The aforementioned derivation can be extended to solutions to the optimization problem (1). Consider a popular setting to alternatively update W and H. If the squared regularization

$$\text{Reg}(W, H) = \lambda \left(\sum_{i=1}^{m} \|\boldsymbol{w}_i\|_2^2 + \sum_{j=1}^{n} \|\boldsymbol{h}_j\|_2^2 \right) \tag{4}$$

is considered, where λ is the regularization hyper-parameter, and the loss in (2) is applied, then the sub-problem of fixing either W or H is a least-square regression problem. Thus, the optimization procedure becomes an alternating least square (ALS) one. It has been proved that one ALS iteration for updating W and H costs

$$O\left(|\Omega^+| k^2 + (m + n) k^3 \right). \tag{5}$$

The details are presented in Supplementary Materials A.1 of [14].

[1] Some works even consider entry-dependent cost α_{ij}, although we use a single value α here for simplicity.

Table 1: Results from [2], where the mean and standard deviation of NDCG@200 and Recall@200 are multiplied by 100. They use SRRMF to denote their ranking-based method of minimizing the sum of (6) and (9). The existing method of using point-wise settings as the loss function is called WRMF, while BPR is the method of using (8).

	MovieLens		Amazon Books		Netflix	
	NDCG@200	Recall@200	NDCG@200	Recall@200	NDCG@200	Recall@200
SRRMF	51.77 ± 0.03	71.65 ± 0.06	9.77 ± 0.03	27.48 ± 0.06	43.97 ± 0.01	59.91 ± 0.03
WRMF	49.74 ± 0.05	69.15 ± 0.07	9.14 ± 0.02	24.52 ± 0.06	42.24 ± 0.01	57.65 ± 0.03
BPR	47.10 ± 0.09	69.91 ± 0.03	6.34 ± 0.09	18.59 ± 0.07	32.92 ± 0.12	51.79 ± 0.06

2.2 Ranking-Based MF by [2]

Chen et al. [2] critiqued the setting in (2), stating that γ_0 is difficult to explain and decide. Therefore, they used a ranking-based loss instead

$$\frac{1}{2}\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\sum_{t\notin\Omega_i^+}\left((1-0)-(\boldsymbol{w}_i^T\boldsymbol{h}_s-\boldsymbol{w}_i^T\boldsymbol{h}_t)\right)^2, \tag{6}$$

where Ω_i^+ includes items observed by user i

$$\Omega_i^+ = \{j \mid r_{ij} = 1\}.$$

From (6), we aim to have

$$1 \approx \boldsymbol{w}_i^T\boldsymbol{h}_s - \boldsymbol{w}_i^T\boldsymbol{h}_t. \tag{7}$$

This means that between an observed (i,s) entry and an unobserved (i,t) entry, $\boldsymbol{w}_i^T\boldsymbol{h}_s$ should be larger than $\boldsymbol{w}_i^T\boldsymbol{h}_t$.[2]

Ranking-based losses have been applied to OCCF (such as in [4, 11]). A popular one is Bayesian Personalized Ranking (BPR) by [13], which puts the ranking difference to a logistic loss.

$$\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\sum_{t\notin\Omega_i^+} -\log\sigma(\boldsymbol{w}_i^T\boldsymbol{h}_s-\boldsymbol{w}_i^T\boldsymbol{h}_t), \text{ where } \sigma(x)=\frac{1}{1+e^{-x}}. \tag{8}$$

Because the summation involves $O(mn)$ entries, the aforementioned problem with a high computational cost is still present. Currently, BPR is primarily trained using stochastic gradient methods, but convergence seems to be slow, as demonstrated in the experiments in [14]. This may be because each stochastic gradient step involves only few (user, item) entries, and many steps are required for a sufficient number of entries to be covered.

Recall that in (2), if a special loss function such as the squared function is used in the summation over all entries, efficient algorithms can be developed to avoid a computational cost proportional to $O(mn)$. Based on this idea, the main contribution of [2] is to consider the squared function in (6) and develop an efficient algorithm. We further note that Chen et al. [2] considered a ranking-based regularization term

$$\mathrm{Reg}(W,H) = \frac{\lambda}{4}\sum_{i=1}^{m}\sum_{s,t\notin\Omega_i^+}(\boldsymbol{w}_i^T\boldsymbol{h}_s-\boldsymbol{w}_i^T\boldsymbol{h}_t)^2. \tag{9}$$

They did so because they thought that "each user's estimated preference for unobserved items should be close to zero, implying some similarity among unobserved items."

By minimizing the sum of (6) and (9), each of which is a squared term, Chen et al. [2] developed an ALS method to alternatively update W and H. The complexity of updating W and H once is the same as (5).

Chen et al. [2] conducted experiments on different one-class MF models. We partially present their results in Table 1. Clearly, their ranking-based approach is consistently the best. The results seem to suggest that to apply one-class MF, one should consider a ranking-based setting. However, we think that this conclusion is unwarranted for the following reasons.

- In the literature of general learning to rank, where objects are in multiple ranks, point-wise and pair-wise approaches are primarily used [1]. A point-wise approach directly approximates the given score by, for example, a regression loss. The setting in (2) is an example. On the other hand, a pair-wise approach aims to rank one entry ahead of another according to their scores, where the setting of (6) is an example. Previous works have shown that in some situations, simple point-wise models are highly competitive (e.g., the discussion in Section 6.3 of [1]). The one-class problem we are handling has only two scores 0 and 1, and it is thus unclear whether the pair-wise ranking-based approach is required.
- Chen et al. [2] stated that the value γ_0 in (2) is difficult to decide. However, this is not the case because this value can often be decided by a validation procedure.
- We find that the design of (6) is different from many other ranking-based losses. In machine learning, if we hope that

$$\boldsymbol{w}_i^T\boldsymbol{h}_s > \boldsymbol{w}_i^T\boldsymbol{h}_t \text{ for any } s\in\Omega_i^+ \text{ and } t\notin\Omega_i^+, \tag{10}$$

the loss function is often designed to satisfy the following property.

$$\text{loss}\begin{cases}>0, & \text{if } \boldsymbol{w}_i^T\boldsymbol{h}_s < \boldsymbol{w}_i^T\boldsymbol{h}_t,\\ \approx 0, & \text{otherwise.}\end{cases} \tag{11}$$

The logistic loss in (8) is an example, while another popular way is the hinge loss

$$\max(0, 1-\boldsymbol{w}_i^T\boldsymbol{h}_s+\boldsymbol{w}_i^T\boldsymbol{h}_t),$$

or its square (squared hinge loss). However, the loss in (6) does not satisfy the property in (11). Instead, it still resembles a point-wise setting through a direct approximation of $\boldsymbol{w}_i^T\boldsymbol{h}_s-\boldsymbol{w}_i^T\boldsymbol{h}_t$ in (7). In Section 3, we mathematically prove that in fact (6) is not very different from the standard point-wise setting in (2).
An interesting question is why Chen et al. [2] did not consider, for example, the squared hinge loss. An apparent reason is that techniques in (3) for reducing the $O(mn)$ computational cost no longer work.

[2] Chen et al. [2] mentioned other ranking-based losses, although they focused on (6).

Table 2: Statistics of data sets.

	Chen et al. [2]			Our reproduction		
Data set	Users	Items	Observed entries	Users	Items	Observed entries
MovieLens	69,838	8,939	9,983,739	69,838	8,939	9,983,739
Amazon Books	158,650	128,939	4,701,968	158,650	128,939	4,701,968
Netflix	463,770	17,764	100,396,329	463,770	17,768	100,396,376

From the above concerns, it is important to re-investigate the experiments in [2].

3 Relation Between Pair-Wise Ranking-Based and Point-Wise Regression-Based MF Models

We mentioned that the loss in (6) differs from other ranking-based losses in that it does not satisfy the property in (10). From the use of a squared function, in this section, we show that (6) is in fact very related to the standard point-wise approach in (2).

To connect (6) to (2), in Appendix 7.1, we derive the following result.

$$(6) = L_p(W,H) + \sum_{i=1}^{m} \sum_{s\in\Omega_i^+} \sum_{t\notin\Omega_i^+} \left(1 - \boldsymbol{w}_i^T\boldsymbol{h}_s\right)\left(\boldsymbol{w}_i^T\boldsymbol{h}_t\right), \quad (12)$$

where

$$L_p(W,H) = \sum_{(i,j)\in\Omega^+} \beta_i(1-\boldsymbol{w}_i^T\boldsymbol{h}_j)^2 + \sum_{(i,j)\notin\Omega^+} \alpha_i(0-\boldsymbol{w}_i^T\boldsymbol{h}_j)^2 \quad (13)$$

is an extension of the point-wise setting (2) with the following coefficients

$$\beta_i = \frac{n-|\Omega_i^+|}{2} \text{ and } \alpha_i = \frac{|\Omega_i^+|}{2},\ \forall i.$$

Now we define $L_r(W,H)$ as the function in (6). We show that an optimal solution of $L_p(W,H)$ may be close to a solution of minimizing $L_r(W,H)$. From (12), we calculate the gradient with respect to $\boldsymbol{w}_i$, $\forall i = 1,\dots,m$.

$$\begin{aligned}&\frac{\partial L_r}{\partial \boldsymbol{w}_i}\\ =&\frac{\partial L_p}{\partial \boldsymbol{w}_i} - \sum_{s\in\Omega_i^+}\boldsymbol{h}_s\cdot\sum_{t\notin\Omega_i^+}\boldsymbol{w}_i^T\boldsymbol{h}_t + \sum_{s\in\Omega_i^+}\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_s\right)\cdot\sum_{t\notin\Omega_i^+}\boldsymbol{h}_t.\end{aligned} \quad (14)$$

If (W^*,H^*) minimizes $L_p(W,H)$, then

$$\frac{\partial L_p(W^*,H^*)}{\partial \boldsymbol{w}_i} = \boldsymbol{0}. \quad (15)$$

Furthermore, from (13), (W^*,H^*) tends to satisfy

$$\left(\boldsymbol{w}_i^*\right)^T\boldsymbol{h}_s^* \approx 1,\ \forall s\in\Omega_i^+ \text{ and } \left(\boldsymbol{w}_i^*\right)^T\boldsymbol{h}_t^* \approx 0,\ \forall t\notin\Omega_i^+. \quad (16)$$

Then, (14), (15), and (16) imply that

$$\frac{\partial L_r(W^*,H^*)}{\partial \boldsymbol{w}_i} \approx \boldsymbol{0}.$$

The situation for the gradient with respect to $\boldsymbol{h}_j$, $j = 1,\dots,n$, is similar. Therefore, (W^*,H^*) may be close to a stationary point of minimizing $L_r(W,H)$. We can thus conjecture that minimizing the point-wise form in (13) leads to a solution that well optimizes the pair-wise form in (6).

4 Reproducing Results in [2]

The discussions in Sections 2 and 3 underscore the importance of rigorously comparing point-wise and pair-wise one-class MFs. Our first step is to investigate experimental settings in [2] and reproduce their results.

4.1 Data Set Generation and Partition

Chen et al. [2] used the data sets "MovieLens" [6], "Amazon Books"[3] and "Netflix"[4] in their experiments. Some modifications are needed for a one-class setting. The procedure was described in their paper, but the processed sets were not available. We carefully followed their descriptions to generate data sets with statistics listed in the right column of Table 2. As a comparison, statistics from [2] are given in the left column of the same table. Clearly, statistics of our generated sets were the same as or close to theirs. Therefore, although sets in [2] were unavailable, we have reasonably reproduced them. More details of the data generation are in Appendix 7.2.

Subsequently, we discuss the partition of each set to training and test subsets because a test set is needed for evaluating a model. Moreover, a machine learning model often involves some hyper-parameters, which are selected by a validation procedure. Therefore, the training set is further split to training and validation sets, and hyper-parameters that lead to the best validation performance are chosen. Chen et al. [2] did not provide details about their data partitions. All they have stated were that "We do cross validation experiment of 5 times on each data set for our method and baselines." Thus, we applied the following procedure.

(1) The data set is randomly split to 90% for training and 10% for testing.
(2) We applied five-fold CV on the training set to select hyper-parameters.

Note that we must address some issues in evaluating validation and test sets; see the description in Section 5.

4.2 Other Settings in [2]

To reproduce their results, we carefully check the following details.

- **Initialization of MF models.** Although the corresponding details are not mentioned in [2], we combed through their code and found that the scaled Gaussian sampling

$$\mathcal{N}(0,1)\times\frac{1}{10}$$

was used on each component of the MF model W and H.

[3] http://jmcauley.ucsd.edu/data/amazon/
[4] https://www.kaggle.com/netflix-inc/netflix-prize-data

Table 3: Reproducing the results in Table 4 of [2]. The pair-wise method by [2] and the point-wise method by solving (17) are called SRRMF and WRMF-(17), respectively.

		SRRMF				WRMF-(17)			
		NDCG@200	Diff.	Recall@200	Diff.	NDCG@200	Diff.	Recall@200	Diff.
MovieLens	Chen et al.	51.77		71.65		49.74		69.15	
	Results on test set	43.57	-8.20	74.58	2.93	42.16	-7.58	73.55	4.40
	CV results on all data	51.34	-0.43	71.54	-0.11	49.78	0.04	72.49	3.34
Amazon Books	Chen et al.	9.77		27.48		9.14		24.52	
	Results on test set	8.31	-1.46	27.68	0.20	7.90	-1.24	24.94	0.42
	CV results on all data	9.73	-0.04	27.32	-0.16	9.18	0.04	24.56	0.04
Netflix	Chen et al.	43.97		59.91		42.24		57.65	
	Results on test set	38.01	-5.96	61.46	1.55	36.01	-6.23	61.58	3.93
	CV results on all data	43.75	-0.22	59.89	-0.02	42.42	0.18	57.99	0.34

- **Evaluation metrics.** Chen et al. [2] considered NDCG@200 and Recall@200. The choice of 200 is surprising because in many practical applications a smaller number of items are recommended to users. Although we followed suit in using a value of 200 in reproducing their results, we also considered NDCG@K and Recall@K with small K values in subsequent experiments.
- **Implementation of the pair-wise method** We use the code[5] of [2] for their method SRRMF of minimizing the sum of (6) and (9). Note that they ran a fixed number of 10 ALS iterations to update W and H in the training process. This choice seems to be arbitrary, but such details are important in our attempt to reproduce their results.
- **The compared point-wise MF model.** Chen et al. [2] denoted the point-wise method used in their experiments as WRMF (weighted regularized MF) and cited the work by [5] in their experiment section. Thus, an optimization problem (2) seems to be considered. Because Chen et al. [2] did not release the implementation of the point-wise method, we communicated with them for further details. Interestingly, they directed us to [8], which is a work in discrete MF. After some investigation,[6] we conclude that the following continuous relaxation is the point-wise method compared in [2]:

$$\min_{\substack{W,H,\\ B_b,D_b,\\ B_d,D_d}} \sum_{(i,j)\in\Omega^+} (1-\boldsymbol{w}_i^T\boldsymbol{h}_j)^2 + \tau_1\left(\|W-B_b\|_F^2 + \|H-D_b\|_F^2\right) + \alpha \sum_{(i,j)\notin\Omega^+} (\boldsymbol{w}_i^T\boldsymbol{h}_j)^2 + \tau_2\left(\|W-B_d\|_F^2 + \|H-D_d\|_F^2\right) \tag{17}$$

subject to

$$B_b^T\mathbf{1}^m = \mathbf{0},\ D_b^T\mathbf{1}^n = \mathbf{0},\ B_d^TB_d = m\cdot\mathcal{I}^k,\ D_d^TD_d = n\cdot\mathcal{I}^k,$$

where $\|\cdot\|_F$ is the Frobenious norm, B_b and $B_d \in \mathbb{R}^{m\times k}$, D_b and $D_d \in \mathbb{R}^{n\times k}$, $\mathbf{1}^m \in \mathbb{R}^m$ and $\mathbf{1}^n \in \mathbb{R}^n$ are vectors containing ones, and $\mathcal{I}^k \in \mathbb{R}^{k\times k}$ is an identity matrix. We follow [2] to use a public code[7] for solving (17). More details are in Appendix 7.3.

- **Optimization method.** We mentioned in Section 2 that Chen et al. [2] derived an ALS procedure to minimize their ranking-based formulation. For the point-wise method of solving (17), an extension of ALS can be considered so that W, H, B_b, D_b, B_d and D_d are sequentially updated. For example, if B_b, D_b, B_d and D_d are fixed, (17) is reduced to the following unconstrained problem that is very close to (2).

$$\min_{W,H} \sum_{(i,j)\in\Omega^+} (1-\boldsymbol{w}_i^T\boldsymbol{h}_j)^2 + \tau_1\left(\|W-B_b\|_F^2 + \|H-D_b\|_F^2\right) + \alpha \sum_{(i,j)\notin\Omega^+} (\boldsymbol{w}_i^T\boldsymbol{h}_j)^2 + \tau_2\left(\|W-B_d\|_F^2 + \|H-D_d\|_F^2\right).$$

Then when W (or H) is fixed, we have a least square problem. Regarding B_b, D_b, B_d or D_d, the situation is more complicated because of constraints. We leave details in Supplementary Materials A.1. Note that like the pair-wise method, Chen et al. [2] also run 10 iterations.
- **Hyper-parameter search.** For their pair-wise approach of using the loss (6) and the regularization term (9), Chen et al. [2] set the latent dimension $k = 32$ and check the following regularization parameters in (9).

$$\lambda \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 5\times10^{-1}\}.$$

For the point-wise MF model in (17), Chen et al. [2] consider $k = 32$ and the following weights of unobserved samples

$$\alpha \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}. \tag{18}$$

However, they did not mention the regularization hyper-parameters τ_1 and τ_2. In the code that we obtained, the default τ_1 and τ_2 were 10^{-2}, but we expanded our scope of consideration to include

$$\tau_1 = \tau_2 \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}.$$

4.3 Reproducing Results in [2]

The goal is to check if results in their Table 4 can be reproduced. To begin, we copy results in [2] as references; see the first row for each data set in Table 3. Next, we conduct a standard machine learning procedure with the hyper-parameter search in Algorithm 1. The performance on predicting the test set is presented in the second row in Table 3. Note that NDCG@200 is the target evaluation metric used in Algorithm 1 for the hyper-parameter selection. In Table 3,

[5] https://github.com/HERECJ/recsys/tree/master/alg/discrete/SRRMF
[6] See details in Appendix 7.3.
[7] https://github.com/DefuLian/recsys/tree/master/alg/discrete/dmf

Table 4: A rigorous comparison between point-wise and pair-wise methods for one-class MF. Performance on a test set is presented. The best result under each evaluation metric is indicated in bold. We consider five random seeds and report the means of the results. The standard deviations and confidence interval figures are located in Appendix 7.4.

		NDCG					Recall				
		@1	@10	@50	@100	@200	@1	@10	@50	@100	@200
MovieLens	SRRMF	**38.80**	32.07	**36.98**	**39.79**	**43.53**	**5.10**	**24.53**	**49.91**	**62.64**	**74.57**
	WRMF-(17)	38.11	31.28	34.16	38.31	42.14	4.94	22.58	48.59	61.43	73.58
	WRMF-(2)	37.18	**32.19**	35.25	39.39	43.15	4.79	23.37	49.77	62.56	74.49
Amazon Books	SRRMF	2.25	3.16	5.37	6.75	8.34	0.86	4.55	13.09	19.46	27.79
	WRMF-(17)	2.22	3.18	5.30	6.52	7.91	0.88	4.56	12.20	18.28	26.36
	WRMF-(2)	**2.50**	**3.49**	**5.87**	**7.23**	**8.76**	**0.93**	**5.04**	**13.70**	**19.96**	**28.03**
Netflix	SRRMF	35.41	28.83	30.18	**33.71**	**38.01**	2.92	15.77	**36.80**	**48.89**	**62.37**
	WRMF-(17)	35.19	28.50	29.51	32.91	36.01	2.85	15.43	35.21	47.98	61.60
	WRMF-(2)	**36.17**	**29.29**	**30.25**	33.63	36.54	**2.94**	**15.87**	35.90	48.73	62.18

Algorithm 1 A standard machine learning procedure.

Given the target evaluation metric.

1° Applying five-fold cross validation (CV) on the training set to select the hyper-parameters achieving the best target evaluation metric.
2° Obtaining the final model by training the whole training set under the selected hyper-parameters.
3° Using the obtained model to predict the test set.

a column "Diff." indicates the difference from the result by [2]. Clearly, our obtained NDCG@200 results are much worse although Recall@200 is generally better. Therefore, it seems that we have failed to reproduce the results in [2].

After some investigation, we suspect that the differences are due to the different training/prediction procedures. Because Chen et al. [2] mentioned only that they "do cross validation experiment of 5 times," we doubt if they have a test set independent of the hyper-parameter search. In other words, they may apply CV on all available data by using various hyper-parameters and report the best CV performance. We consider this setting and report results in Table 3; see the row of "CV results on all data." Interestingly, the resulting NDCG@200 and Recall@200 are very close to those in [2]. Therefore, after carefully handling many details, we are able to reproduce their results. However, through this process we have identified several possible issues in their experiments.

- They used (17) as the point-wise method for comparison, but in the literature the most widely used setting is to minimize (2).
- They may have reported the validation results after tuning hyper-parameters. This setting is inappropriate because results on an independent test should be used in comparing two methods.

5 A Rigorous Comparison Between Point-Wise and Pair-Wise One-Class MFs

By applying suitable settings, in this section we rigorously compare point-wise and pair-wise one-class MFs. The methods included in our experiments are described as follows. One is a pair-wise method, while two are point-wise methods.

- SRRMF: This is the pair-wise method proposed in [2], which minimizes the sum of (6) and (9).
- WRMF-(17): As we concluded via experiments in Section 4, this method seems to be the point-wise approach compared in [2]. Obviously, it should be included in our experiments.
- WRMF-(2): We learned earlier that WRMF-(17) is not the most commonly used point-wise method for one-class MF because of the constraints. Therefore, we consider the more popular form of minimizing (2), which has been well studied in, for example, [10, 14]. The implementation is modified from the code[8] by [14]. The hyper-parameters of this method are α, λ and γ_0. For the search process, we consider α in the set in (18),

$$\lambda \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\},$$

and

$$\gamma_0 \in \{0, 10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}\}.$$

For each of the above approaches, various optimization methods may be applied. To avoid the effect of optimization techniques, we solve all problems by the alternating least square (ALS) method. Moreover, we follow [2] to run 10 ALS iterations for each method.

About the evaluation metrics, Chen et al. [2] considered NDCG@K and Recall@K with $K = 200$. We mentioned in Section 4.2 that in evaluating a practical recommender system, a smaller K is often more suitable. Thus here we consider a wide range of K values:

$$1, 10, 50, 100, 200.$$

An important but often ignored detail in calculating a metric is that training data should be excluded to avoid an overestimation. Because the model was obtained by fitting Ω_i^+, the prediction on elements in Ω_i^+ should be quite accurate. Therefore, we follow [14] to exclude Ω_i^+ and use only predicted values in

$$\{1, \dots, n\} \setminus \Omega_i^+$$

for calculating the metric.

We then rigorously apply the machine learning procedure in Algorithm 1. In particular, each evaluation metric requires a corresponding CV process. This setting is laborious, but mimics the practical use, where a user decides a suitable metric and applies CV to find the best hyper-parameters.

[8] https://www.csie.ntu.edu.tw/~cjlin/papers/one-class-mf/

To address the variance from different data-set partitions for cross validation and training/test splits, we consider five random seeds. The means of the results are shown in Table 4, and the standard deviations and confidence interval figures are located in Appendix 7.4. We can make the following observations.

- For the evaluation metrics NDCG@200 and Recall@200 considered by [2], their method SRRMF is better than WRMF-(17). The superiority of their method is consistent with their experimental results, which are shown in Table 1. However, the gap is now generally smaller. Next, in the comparison with the other point-wise method, WRMF-(2), the pair-wise method is not always better. Even if it is, the difference is usually very small. Because WRMF-(2) is the most popular point-wise setting, this result seems to contradict the conclusion by [2] in comparing point-wise and pair-wise one-class MFs.
- If a smaller K is considered in NDCG@K and Recall@K, the pair-wise setting is often worse than the point-wise setting WRMF-(2). Take $K = 10$ for an example. WRMF-(2) is superior in almost all cases. Because $K = 10$ is more often used than $K = 200$ in real applications, our experiments suggest that in practice we should consider the point-wise setting in (2) first.
- From Table 7 in Appendix 7.5, results of using $\gamma_0 = 0$ are almost the same as those in Table 4, where γ_0 has been tuned. Therefore, the selection of γ_0 seems to be easy. This observation is opposite to the claim in [2] on the difficulty in deciding γ_0.

6 Conclusions

In this paper, we re-investigate the conclusion made by [2] on the superiority of pair-wise ranking-based one-class MF over point-wise settings. By mathematical derivations, we explain that their method may perform only similar to point-wise ones. Further, through the process to reproduce their experimental results, we identify some possible issues in their settings. For example, Chen et al. [2] seem to report cross validation results after hyper-parameter tuning, but for performance evaluation, an independent test set should be used. More importantly, the point-wise method employed for the comparison is not the most commonly used one. After considering suitable settings, we rigorously compare point-wise and pair-wise one-class MFs. Our results show that the pair-wise method, if not inferior, is only similar to the point-wise setting. Therefore, for one-class MF, the more traditional and mature point-wise setting should still be the method for consideration. Our findings not only contradict the conclusions in [2], but also indicate the importance of carefully considering details in comparing two machine learning methods.

7 Appendices

7.1 The Derivation of (12)

Let us start from (6) $- L_p(W, H)$. Note that the coefficients

$$\beta_i = \frac{n - |\Omega_i^+|}{2} \text{ and } \alpha_i = \frac{|\Omega_i^+|}{2}, \ \forall i,$$

and details of $L_p(W, H)$ are in (13).

$$\begin{aligned}
&(6) - (13)\\
=&\frac{1}{2}\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\sum_{t\notin\Omega_i^+}\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_s+\boldsymbol{w}_i^T\boldsymbol{h}_t\right)^2-\sum_{(i,j)\in\Omega^+}\beta_i\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_j\right)^2\\
&-\sum_{(i,j)\notin\Omega^+}\alpha_i\left(\boldsymbol{w}_i^T\boldsymbol{h}_j\right)^2\\
=&\frac{1}{2}\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\sum_{t\notin\Omega_i^+}\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_s+\boldsymbol{w}_i^T\boldsymbol{h}_t\right)^2-\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\beta_i\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_j\right)^2\\
&-\sum_{i=1}^{m}\sum_{t\notin\Omega_i^+}\alpha_i\left(\boldsymbol{w}_i^T\boldsymbol{h}_j\right)^2\\
=&\frac{1}{2}\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\sum_{t\notin\Omega_i^+}\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_s\right)^2+\frac{1}{2}\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\sum_{t\notin\Omega_i^+}\left(\boldsymbol{w}_i^T\boldsymbol{h}_t\right)^2\\
&+\frac{1}{2}\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\sum_{t\notin\Omega_i^+}2\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_s\right)\left(\boldsymbol{w}_i^T\boldsymbol{h}_t\right)-\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\beta_i\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_j\right)^2\\
&-\sum_{i=1}^{m}\sum_{t\notin\Omega_i^+}\alpha_i\left(\boldsymbol{w}_i^T\boldsymbol{h}_j\right)^2\\
=&\frac{1}{2}\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}2\beta_i\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_s\right)^2+\frac{1}{2}\sum_{i=1}^{m}\sum_{t\notin\Omega_i^+}2\alpha_i\left(\boldsymbol{w}_i^T\boldsymbol{h}_t\right)^2\\
&+\frac{1}{2}\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\sum_{t\notin\Omega_i^+}2\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_s\right)\left(\boldsymbol{w}_i^T\boldsymbol{h}_t\right)-\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\beta_i\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_j\right)^2\\
&-\sum_{i=1}^{m}\sum_{t\notin\Omega_i^+}\alpha_i\left(\boldsymbol{w}_i^T\boldsymbol{h}_j\right)^2\\
=&\sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\sum_{t\notin\Omega_i^+}\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_s\right)\left(\boldsymbol{w}_i^T\boldsymbol{h}_t\right).
\end{aligned}$$

With moving (13) to the right-hand side, we have done the derivation

$$(6) = (13) + \sum_{i=1}^{m}\sum_{s\in\Omega_i^+}\sum_{t\notin\Omega_i^+}\left(1-\boldsymbol{w}_i^T\boldsymbol{h}_s\right)\left(\boldsymbol{w}_i^T\boldsymbol{h}_t\right),$$

which is the same as (12).

7.2 The Processing of Data Sets

The data sets MovieLens, Amazon Books and Netflix are collected as the rating of user i on item j, but the OCCF problem deals with a binary situation of whether user i likes item j. Thus, Chen et al. [2] mapped the rating values in the range $[0, 5]$ to $\{0, 1\}$ by the following function

$$T(v) = \begin{cases} 1 & \text{if } v > 0, \\ 0 & \text{if } v = 0. \end{cases}$$

Furthermore, Chen et al. [2] alternately filtered the users and items with less than 20, 10 and 10 observed entries in MovieLens, Amazon Books and Netflix, respectively. This filtering process, denoted as a

Table 5: The results of two possible point-wise one-class MF models that Chen et al. [2] might use. Values of NDCG@200 and Recall@200 are multiplied by 100. We show CV results after the hyper-parameter search.

	MovieLens		Amazon Books		Netflix	
	NDCG@200	Recall@200	NDCG@200	Recall@200	NDCG@200	Recall@200
Chen et al.	49.74	69.15	9.14	24.52	42.24	57.65
Discrete MF-(19)	32.61	55.31	4.94	15.47	22.68	37.09
WRMF-(17)	**49.78**	**72.49**	**9.18**	**24.56**	**42.42**	**57.99**

Table 6: Results of the same experiment for Table 4, but this table further includes the standard deviations.

		NDCG					Recall				
		@1	@10	@50	@100	@200	@1	@10	@50	@100	@200
MovieLens	SRRMF	**38.80**	32.07	**36.98**	**39.79**	**43.53**	**5.10**	**24.53**	**49.91**	**62.64**	**74.57**
		± 0.12	± 0.09	± 0.04	± 0.07	± 0.08	± 0.04	± 0.09	± 0.03	± 0.04	± 0.04
	WRMF-(17)	38.11	31.28	34.16	38.31	42.14	4.94	22.58	48.59	61.43	73.58
		± 0.19	± 0.06	± 0.04	± 0.05	± 0.05	± 0.03	± 0.06	± 0.07	± 0.02	± 0.03
	WRMF-(2)	37.18	**32.19**	35.25	39.39	43.15	4.79	23.37	49.77	62.56	74.49
		± 0.22	± 0.05	± 0.04	± 0.04	± 0.04	± 0.02	± 0.08	± 0.05	± 0.05	± 0.06
Amazon Books	SRRMF	2.25	3.16	5.37	6.75	8.34	0.86	4.55	13.09	19.46	27.79
		± 0.06	± 0.05	± 0.04	± 0.05	± 0.06	± 0.04	± 0.05	± 0.10	± 0.13	± 0.18
	WRMF-(17)	2.22	3.18	5.30	6.52	7.91	0.88	4.56	12.20	18.28	26.36
		± 0.05	± 0.04	± 0.05	± 0.05	± 0.06	± 0.02	± 0.05	± 0.12	± 0.16	± 0.15
	WRMF-(2)	**2.50**	**3.49**	**5.87**	**7.23**	**8.76**	**0.93**	**5.04**	**13.70**	**19.96**	**28.03**
		± 0.06	± 0.04	± 0.04	± 0.05	± 0.05	± 0.03	± 0.08	± 0.07	± 0.15	± 0.15
Netflix	SRRMF	35.41	28.83	30.18	**33.71**	**38.01**	2.92	15.77	**36.80**	**48.89**	**62.37**
		± 0.05	± 0.04	± 0.03	± 0.03	± 0.03	± 0.01	± 0.04	± 0.01	± 0.02	± 0.02
	WRMF-(17)	35.19	28.50	29.51	32.91	36.01	2.85	15.43	35.21	47.98	61.60
		± 0.08	± 0.04	± 0.04	± 0.04	± 0.03	± 0.02	± 0.03	± 0.03	± 0.03	± 0.02
	WRMF-(2)	**36.17**	**29.29**	**30.25**	33.63	36.54	**2.94**	**15.87**	35.90	48.73	62.18
		± 0.07	± 0.05	± 0.05	± 0.05	± 0.01	± 0.02	± 0.06	± 0.02	± 0.02	± 0.06

function F, is repetitively applied.

$$\text{data\_set}' = \underbrace{F \circ \cdots \circ F}_{n}(\text{data\_set}),$$

where n is a large enough number so that

$$F(\text{data\_set}') = \text{data\_set}'.$$

We follow the same procedure to process each set and list statistics of the processed sets in the right column of Table 2.

7.3 The Point-Wise One-Class MF Used in [2]

From the communication with Chen et al. [2], we learned that their experiment code of WRMF is at https://github.com/HERECJ/recsys/tree/master/alg/discrete/dmf. However, after our investigation, this code is from the work [8] and solves the following integer programming problem.

$$\begin{aligned}\min_{\substack{W,H,\\ B_b,D_b,\\ B_d,D_d}} & \sum_{(i,j)\in\Omega^+} (1 - \boldsymbol{w}_i^T \boldsymbol{h}_j)^2 + \tau_1 \left(\|W - B_b\|_F^2 + \|H - D_b\|_F^2 \right) \\ & + \alpha \sum_{(i,j)\notin\Omega^+} (\boldsymbol{w}_i^T \boldsymbol{h}_j)^2 + \tau_2 \left(\|W - B_d\|_F^2 + \|H - D_d\|_F^2 \right) \end{aligned} \tag{19}$$

subject to

$$B_b^T \mathbf{1}^m = \mathbf{0},\ D_b^T \mathbf{1}^n = \mathbf{0},\ B_d^T B_d = m \cdot \mathcal{I}^k,\ D_d^T D_d = n \cdot \mathcal{I}^k,$$
$$W \in \{-1, 1\}^{m \times k}, H \in \{-1, 1\}^{n \times k},$$

which is denoted as "Discrete MF-(19)." Furthermore, Lian et al. [8] initialized the solution procedure of (19) by solving a continuous relaxation problem (17), which is referred to as "WRMF-(17)," so there are two point-wise implementations in their code. Because we do not exactly know between "Discrete MF-(19)" and "WRMF-(17)," which one is used by [2], we conduct an experiment in Table 5. We present CV results after searching hyper-parameters. Therefore, results in the third row are the same as those in the row of "CV results on all data" in Table 3. See details about the experimental settings in Section 4. From Table 5, clearly, results of WRMF-(17) are closer to those reported in [2]. Therefore, we conclude that the

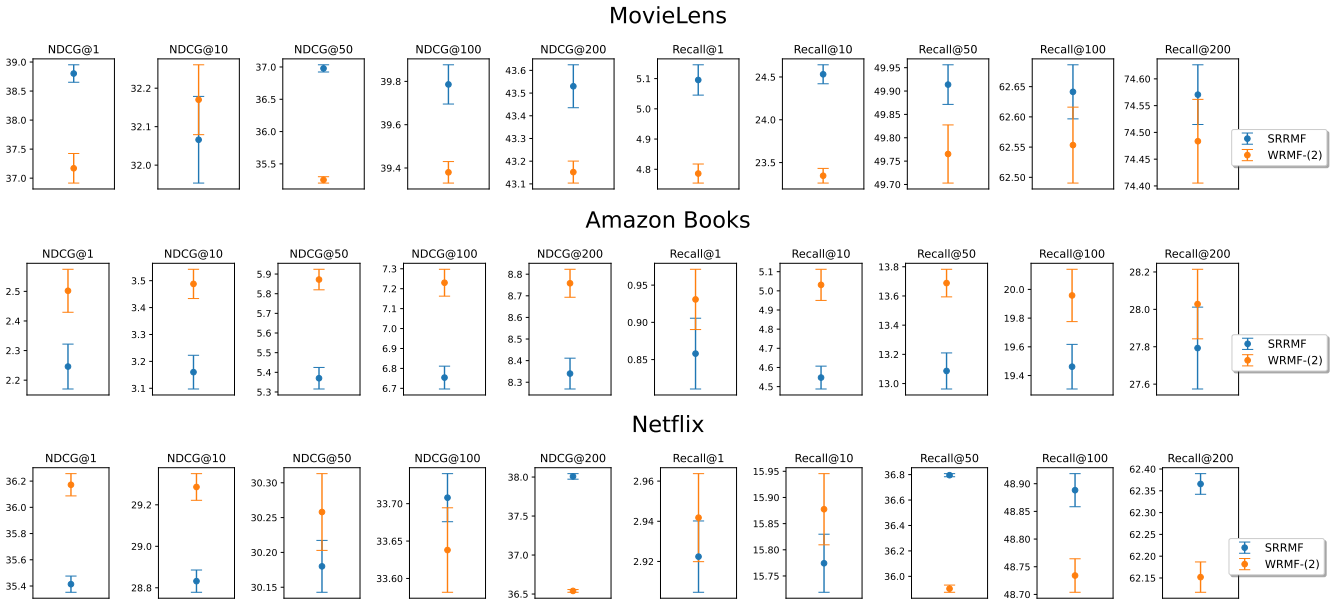

Figure 1: The 95% confidence interval for the results of SRRMF and WRMF-(2).

Table 7: Results of the same experiment for Table 4, but $\gamma_0 = 0$ is fixed without tuning.

		NDCG					Recall				
		@1	@10	@50	@100	@200	@1	@10	@50	@100	@200
MovieLens	SRRMF	**38.80**	32.07	**36.98**	**39.79**	**43.53**	**5.10**	**24.53**	**49.91**	**62.64**	**74.57**
	WRMF-(17)	38.11	31.28	34.16	38.31	42.14	4.94	22.58	48.59	61.43	73.58
	WRMF-(2)	37.17	**32.17**	35.25	39.38	43.15	4.79	23.35	49.77	62.55	74.48
Amazon Books	SRRMF	2.25	3.16	5.37	6.75	8.34	0.86	4.55	13.09	19.46	27.79
	WRMF-(17)	2.22	3.18	5.30	6.52	7.91	0.88	4.56	12.20	18.28	26.36
	WRMF-(2)	**2.50**	**3.49**	**5.87**	**7.23**	**8.76**	**0.93**	**5.03**	**13.69**	**19.96**	**28.03**
Netflix	SRRMF	35.41	28.83	30.18	**33.71**	**38.01**	2.92	15.77	**36.80**	**48.89**	**62.37**
	WRMF-(17)	35.19	28.50	29.51	32.91	36.01	2.85	15.43	35.21	47.98	61.60
	WRMF-(2)	**36.17**	**29.29**	**30.26**	33.64	36.54	**2.94**	**15.88**	35.90	48.73	62.15

setting of solving (17) should be the point-wise approach used in [2].

7.4 Confidence Interval for Results

Due to the readability, we only show the mean of results under different random seeds in Section 5. Complete results including the standard deviation are presented in Table 6. We then calculate the confidence interval with Student's t distribution with 4 degrees of freedom, which is

$$[\text{mean} - 1.2415 \cdot \text{std}, \text{mean} + 1.2415 \cdot \text{std}],$$

and plot the intervals of results of SRRMF and WRMF-(2) in Figure 1. We can observe that standard deviations are generally too small to affect our conclusions on the comparison.

7.5 Results of Using $\gamma_0 = 0$

In Section 2.2, we mentioned that Chen et al. [2] criticized the setting in (2) because they think γ_0 is difficult to explain and decide. Nevertheless, we think γ_0 can often be decided by a validation procedure; see results in Table 4. Moreover, if we fix $\gamma_0 = 0$, results shown in Table 7 are almost the same as those in Table 4. Thus, deciding a suitable γ_0 seems to be easy.

Acknowledgments

We appreciate the authors of [2] for responding to our questions while reproducing their work. This article was subsidized for English editing by National Taiwan University under the Excellence Improvement Program for Doctoral Students (grant number 108-2926-I-002-002-MY4), sponsored by National Science and Technology Council, Taiwan. This work was supported by National Science and Technology Council of Taiwan grant 110-2221-E-002-115-MY3.

References

[1] Olivier Chapelle and Yi Chang. 2011. Yahoo! Learning to Rank Challenge Overview. In *JMLR Workshop and Conference Proceedings: Workshop on Yahoo! Learning to Rank Challenge*, Vol. 14. 1–24.

[2] Jin Chen, Defu Lian, and Kai Zheng. 2019. Improving one-class collaborative filtering via ranking-based implicit regularizer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 37–44.
[3] Lars Eldén and Haesun Park. 1999. A Procrustes problem on the Stiefel manifold. *Numer. Math.* 82, 4 (1999), 599–619.
[4] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web (WWW)*. 507–517.
[5] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*. 263–272.
[6] Noam Koenigstein and Ulrich Paquet. 2013. Xbox movies recommendations: Variational Bayes matrix factorization with embedded feature selection. In *Proceedings of the 7th ACM Conference on Recommender Systems*. 129–136.
[7] Yanen Li, Jia Hu, Chengxiang Zhai, and Ye Chen. 2010. Improving one-class collaborative filtering by incorporating rich user information. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM)*. 959–968.
[8] Defu Lian, Xing Xie, and Enhong Chen. 2019. Discrete matrix factorization and extension for fast item recommendation. *IEEE Transactions on Knowledge and Data Engineering* 33, 5 (2019), 1919–1933.
[9] Rong Pan and Martin Scholz. 2009. Mind the Gaps: Weighting the Unknown in Large-scale One-class Collaborative Filtering. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. 667–676.
[10] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-class collaborative filtering. In *IEEE International Conference on Data Mining (ICDM)*. 502–511.
[11] Weike Pan and Li Chen. 2013. GBPR: Group preference based Bayesian personalized ranking for one-class collaborative filtering. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI)*. 2691–2697.
[12] Ulrich Paquet and Noam Koenigstein. 2013. One-class collaborative filtering with random graphs. In *Proceedings of the 22nd International Conference on World Wide Web*. 999–1008.
[13] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI)*. 452–461.
[14] Hsiang-Fu Yu, Mikhail Bilenko, and Chih-Jen Lin. 2017. Selection of Negative Samples for One-class Matrix Factorization. In *Proceedings of SIAM International Conference on Data Mining (SDM)*. http://www.csie.ntu.edu.tw/~cjlin/papers/one-class-mf/biased-mf-sdm-with-supp.pdf

A Supplementary Materials

A.1 Extending ALS for Solving (17)

To solve (17), we mentioned in Section 4.2 that ALS is extended to update W, H, B_b, D_b, B_d and D_d sequentially. We showed that the sub-problem of W or H is a standard least square problem and can be easily solved. Here we discuss how to update B_b and B_d by the following derivation, while the situation for D_b and D_d is similar.

If all variables except B_b are fixed, the sub-problem is

$$\begin{aligned} &\min_{B_b} \|W - B_b\|_F^2 \\ &\text{subject to } B_b^T \mathbf{1}^m = \mathbf{0}. \end{aligned} \tag{20}$$

Lian et al. [8] stated only that they solve (20) by the Lagrangian multiplier method, but did not give details. Here we derive the solution procedure. Let us modify (20) to another formulation

$$\begin{aligned} &\min_{\hat{\boldsymbol{b}}_1^b,\ldots,\hat{\boldsymbol{b}}_k^b} \sum_{j=1}^{k} \|\hat{\boldsymbol{w}}_j - \hat{\boldsymbol{b}}_j^b\|_2^2 \\ &\text{subject to } \left(\hat{\boldsymbol{b}}_j^b\right)^T \mathbf{1}^m = 0,\ \forall j = 1,\ldots,k, \end{aligned} \tag{21}$$

where $\hat{\boldsymbol{w}}_j$ is the jth column of W and $\hat{\boldsymbol{b}}_j^b$ is the jth column of B_b. Then, we solve (21) by the Lagrangian multiplier method. With the Lagrangian multiplier

$$\boldsymbol{\eta} = \begin{bmatrix} \eta_1 \\ \vdots \\ \eta_k \end{bmatrix} \in \mathbb{R}^k,$$

we have

$$\frac{\partial \sum_{j=1}^{k} \|\hat{\boldsymbol{w}}_j - \hat{\boldsymbol{b}}_j^b\|_2^2}{\partial \hat{\boldsymbol{b}}_i^b} = \sum_{j=1}^{k} \eta_j \cdot \frac{\partial \left(\hat{\boldsymbol{b}}_j^b\right)^T \mathbf{1}^m}{\partial \hat{\boldsymbol{b}}_i^b},$$

for all $i = 1,\ldots,m$, which then implies

$$\frac{\partial \|\hat{\boldsymbol{w}}_i - \hat{\boldsymbol{b}}_i^b\|_2^2}{\partial \hat{\boldsymbol{b}}_i^b} = \eta_i \cdot \frac{\partial \left(\mathbf{1}^m\right)^T \hat{\boldsymbol{b}}_i^b}{\partial \hat{\boldsymbol{b}}_i^b},\ \forall i. \tag{22}$$

Therefore, any optimal solution of (21) satisfies (22) and

$$\left(\mathbf{1}^m\right)^T \hat{\boldsymbol{b}}_1^b = \cdots = \left(\mathbf{1}^m\right)^T \hat{\boldsymbol{b}}_k^b = 0. \tag{23}$$

Further,

$$\begin{aligned} (22) &\Rightarrow -2\hat{\boldsymbol{w}}_i + 2\hat{\boldsymbol{b}}_i^b - \eta_i \mathbf{1}^m = \mathbf{0},\ \forall i \\ &\Rightarrow \hat{\boldsymbol{b}}_i^b = \hat{\boldsymbol{w}}_i + \frac{1}{2}\eta_i \mathbf{1}^m,\ \forall i \\ &\Rightarrow \left(\mathbf{1}^m\right)^T \hat{\boldsymbol{b}}_i^b = \left(\mathbf{1}^m\right)^T \hat{\boldsymbol{w}}_i + \frac{1}{2}\eta_i \left(\mathbf{1}^m\right)^T \mathbf{1}^m,\ \forall i \end{aligned} \tag{24}$$

From (23), we have

$$0 = \left(\mathbf{1}^m\right)^T \hat{\boldsymbol{w}}_i + \frac{m}{2}\eta_i,\ \forall i.$$

By substituting η_i in (24) with the above value, the optimal $\left(\hat{\boldsymbol{b}}_i^b\right)^*$ is

$$\left(\hat{\boldsymbol{b}}_i^b\right)^* = \hat{\boldsymbol{w}}_i - \frac{1}{m}\left(\left(\mathbf{1}^m\right)^T \hat{\boldsymbol{w}}_i\right) \cdot \mathbf{1}^m,\ \forall i.$$

This can be written in a matrix form

$$B_b^* = \left(\mathcal{I}^m - \frac{1}{m}\mathbf{1}^m \left(\mathbf{1}^m\right)^T\right) W,$$

which is the solution of the sub-problem (20).

Now if all variables except B_d are fixed, the sub-problem is

$$\begin{aligned} &\min_{B_d} \|W - B_d\|_F^2 \\ &\text{subject to } B_d^T B_d = m \cdot \mathcal{I}^k. \end{aligned} \tag{25}$$

In Lian et al. [8], they only stated that "this problem is the same as the projection of a matrix onto the Stiefel manifold" and refer to [3] for getting the analytical solution of B_d. Here we derive all the details. The objective function can further be derived as

$$\begin{aligned} &\|W - B_d\|_F^2 \\ &= \sum_i \|\boldsymbol{w}_i - \boldsymbol{b}_i^d\|_2^2 \\ &= \sum_i \|\boldsymbol{w}_i\|_2^2 - 2\sum_i \boldsymbol{w}_i^T \boldsymbol{b}_i^d + \sum_i \|\boldsymbol{b}_i^d\|_2^2 \\ &= \sum_i \|\boldsymbol{w}_i\|_2^2 - 2 \cdot \text{trace}(W^T B_d) + \text{trace}(B_d^T B_d), \end{aligned}$$

where $\boldsymbol{w}_i$ is the ith column of W, and $\boldsymbol{b}_i^d$ is the ith column of B_d. From the constraint in (25),

$$\text{trace}(B_d^T B_d) = \text{trace}(m \cdot \mathcal{I}^k) = mk,$$

so problem (25) is equivalent to

$$\begin{aligned} &\max_{B_d} \text{trace}(W^T B_d) \\ &\text{subject to } B_d^T B_d = m \cdot \mathcal{I}^k. \end{aligned} \tag{26}$$

The maximal B_d can be found by calculating the singular value decomposition (SVD) of W. Suppose that

$$W = U\Sigma V^T,$$

where

$$U \in \mathbb{R}^{m\times m}, \Sigma \in \mathbb{R}^{m\times k}, V \in \mathbb{R}^{k\times k},$$

is the SVD of W with

$$UU^T = U^T U = \mathcal{I}^m \text{ and } VV^T = V^T V = \mathcal{I}^k. \tag{27}$$

Furthermore, because $k \le m$, Σ is a rectangular diagonal matrix having values in the upper $k \times k$ part.

We define Z by

$$Z = \frac{1}{\sqrt{m}} U^T B_d V \in \mathbb{R}^{m\times k}. \tag{28}$$

The objective value of (25) can be written as

$$\begin{aligned} &\text{trace}\left(W^T B_d\right) = \text{trace}\left(\left(U\Sigma V^T\right)^T B_d\right) \\ =&\text{trace}\left(V\Sigma^T U^T B_d\right) = \sqrt{m} \cdot \text{trace}\left(V\Sigma^T Z V^T\right), \end{aligned} \tag{29}$$

where the last equality is from multiplying with VV^T, (27), and (28). The property

$$\text{trace}\,(AB) = \text{trace}\,(BA)\,,\ \forall A, B \in \mathbb{R}^{k\times k}$$

implies that

$$\sqrt{m} \cdot \text{trace}\left(V\Sigma^T Z V^T\right) = \sqrt{m} \cdot \text{trace}\left(\Sigma^T Z V^T V\right)$$
$$= \sqrt{m} \cdot \text{trace}\left(\Sigma^T Z\right) = \sqrt{m} \cdot \sum_{i=1}^{k} \sigma_i \cdot z_{ii}, \tag{30}$$

where σ_i is the ith diagonal component of Σ, z_{ij} is the component on ith row and jth column of Z. Because (27), (28) and the constraint in (26) imply

$$Z^T Z = \frac{1}{m} V^T B_d^T U \cdot U^T B_d V = I^k,$$

we have the property

$$\sum_{s=1}^{m} z_{si}^2 = 1, \ \forall i = 1, \ldots, k,$$

which implies

$$|z_{ii}| \leq 1, \ \forall i.$$

Therefore, from (29) and (30),

$$\text{trace}\left(W^T B_d\right) = \sqrt{m} \cdot \sum_{i=1}^{k} \sigma_i \cdot z_{ii} \leq \sqrt{m} \cdot \sum_{i=1}^{k} \sigma_i,$$

and the equality holds under the condition

$$z_{ii} = 1, \ \forall i = 1, \ldots, k. \tag{31}$$

By considering

$$B_d^* = \sqrt{m} \tilde{U} V^T, \tag{32}$$

where $\tilde{U}$ includes the first k columns of U, and the property (27), we have

$$\left(B_d^*\right)^T B_d^* = m \cdot I^k.$$

Thus B_d^* is feasible to (25). Moreover, from (28), we have

$$Z^* = \frac{1}{\sqrt{m}} U^T B_d^* V = \begin{bmatrix} I^k \\ \mathbf{0} \end{bmatrix}$$

satisfying (31). Therefore, the maximal possible objective value of (26) is achieved. That is, from (29), (30) and (31), we have

$$\text{trace}(W^T B_d) \leq \text{trace}(W^T B_d^*) = \sqrt{m} \cdot \sum_{i=1}^{k} \sigma_i,$$

for any B_d satisfying the constraint of (25). Thus B_d^* in (32) is an optimal solution of (25).