Neurocomputing 275 (2018) 2019–2030

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Effects of preprocessing and training biases in latent factor models for recommender systems

Ye Yuan[1], Xin Luo[1,*], Ming-Sheng Shang[**]

Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China

ARTICLE INFO

Article history:
Received 16 July 2017
Revised 18 September 2017
Accepted 27 October 2017
Available online 11 November 2017

Communicated by Dr. Nianyin Zeng

Keywords:
Recommender system
Latent factor model
Linear bias
High-dimensional and sparse matrix
Neural network

ABSTRACT

Latent factor (LF)-based models are highly efficient in addressing high-dimensional and sparse (HiDS) matrices raised in big-data-related applications like recommender systems. While linear biases have proven to be effective in improving the prediction accuracy and computational efficiency of LF models, their individual and combinational effects in such performance gain remain unclear. To address this issue, this work aims at studying the effects of linear biases in LF models for recommender systems. Based on careful investigations into existing methods, we categorize frequently adopted biases into two classes, i.e., preprocessing bias (PB) and training bias (TB). Subsequently, we deduce the training objectives and parameter updating rules of LF models with different PB and TB combinations. Experimental results on three HiDS matrices generated by real recommender systems show that (a) each PB/TB does have positive/negative effects in the performance of an LF model; (b) Such effects are partially data dependent, however, some specific PB/TB can bring stable performance gain to an LF model; and (c) several PB and TB combinations appear significantly more effective in improving an LF model's performance when compared to their peers.

1. Introduction

The rapid growth of the Internet has brought human society into a fast developing age of information, making online consumption indispensable in people's daily life. Millions of products are provided by numerous online retailers, providing people with various consuming choices. On the other hand, such massive information leads to the problem of information overload, making people be inundated with choices. Hence, recommender systems, which connect people with their potential preferences based on their historical behaviors, become highly desirable [1–12].

In general, recommender systems fall into two categories: content-based [8–12] recommenders and collaborative filtering (CF)-based [12–15] recommenders. Owing to their computational efficiency and effectiveness in generating recommendations, CF-based recommenders have been widely investigated and adopted during the past decade [12–17].

A CF-based recommender commonly models the historical user data (either explicit ratings, or implicit behaviors such as purchasing history, browsing history and searching history) related to involved items into a user-item rating matrix, where high ratings usually indicate users' strong preferences on corresponding items. As user and item numbers explosively increase, it becomes impossible for a user to review all items. Hence, in a CF-based recommender systems, the resultant rating matrix is high-dimensional and sparse (HiDS) with numerous missing data denoting users' unobserved preferences. On the other hand, once we predict these missing data with high accuracy based on the known ones, the system would be able to connect people with their potential favorites.

Thus, for a CF-based recommender, the problem of personalized recommendation is transformed into missing data estimation, where the key is to estimate the unknown entries in its rating matrix based on the known ones. According to recent progress in recommender systems [17], current CF-based recommenders generally can be divided into two branches, i.e., the neighborhood-based models [18,19] and latent factor (LF) models [20–24]. Specifically, a neighborhood-based model works by establishing the relationships among items or users, and estimates unknown ratings based on known ones attached to their K-nearest-neighbors [25].

* Corresponding author at: Chongqing Key Laboratory of Big Data and Intelligent Computing, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China

** Corresponding author.

E-mail addresses: yuanye@cigit.ac.cn (Y. Yuan), luoxin21@cigit.ac.cn (X. Luo), msshang@cigit.ac.cn (M.-S. Shang).

1 Ye Yuan and Xin Luo contributed equally to this work.

https://doi.org/10.1016/j.neucom.2017.10.040
0925-2312/© 2017 Elsevier B.V. All rights reserved.

On the other hand, an LF model-based recommender maps both users and items into the same LF space where a user/an item is connected with an LF vector inferred from the existing ratings. Predictions for unknown ratings are generated heavily relying on these obtained LFs. Compared with neighborhood-based models, a LF model-based recommender [26,27] usually is (a) more accurate in predicting missing data in the target matrix, and (b) highly efficient with low computational and storage costs. Hence, LF model-based recommenders have been frequently mentioned and investigated in the recommender systems community [28], as well as been popular in industrial applications [22–27].

According to prior works, different users or items have differences in the rating scale, which is also known as individual bias [22–27]. For example, typical collaborative filtering data exhibits significant tendencies for some users to give higher ratings than others [22–27]. Thus, extending an LF model recommender by incorporating linear biases (LBs) describing such 'biased effects' into it is a frequently adopted strategy, which is able to improve its prediction accuracy as well as convergence rate.

In general, two kinds of LBs are often encountered in LF models. The first kind of LBs act similarly to the principle factors in artificial neural networks (ANN) [29,30]. Since such a bias will be trained along with the desired LFs, we call it the training bias (TB). TB is frequently seen in sophisticated LF models, like the biased, regularized matrix factorization model [19], singular value decomposition plus-plus model [23], probabilistic matrix factorization model [32], and non-negative LF models [33,34]. The second kind of LBs is acquired based on the statistics of the known data in the target matrix. Since such an LB is pre-computed based on the prior information before the training process of an LF model [22,23], we call it preprocessing bias (PB).

Although current works illustrate that incorporating LBs into an LF model is effective in improving its performance, the individual and combinational effects of each integrated LB in such performance gain remain unclear. For instance, considering adopting TBs, existing works [19,23,32–34] all adopt user and item TBs simultaneously without further study regarding the separate and combined effects of either user or item TB. To the authors' best knowledge, systematically empirical study regarding this issue is not encountered in previously study. Note that different LBs reflect different user/item features in practice. For instance, a user's PB usually reflects his/her rating patterns, while TB is the principle LF hidden in his/her known ratings. Hence, it is highly interesting to seek for the answers to the following questions:

(a) Whether TBs and PBs can work compatibly to achieve a more powerful LF model? and
(b) What are the separate and combined effects of TBs and PBs?

This work investigates the effects by TBs and PBs in an LF model thoroughly and systematically. With such efforts, it aims at providing evidence and guideline for researchers and engineers who want to implement an LF model enhanced by LBs. The contributions of this work include:

(a) *Theories:* According to the incorporation of different combinations of TBs and PBs, the parameter update rules and training processes of an LF model with Euclidean distance as loss function is carefully studied to achieve their general mathematic expressions;
(b) *Empirical studies:* Thoroughly and systematically empirical studies are conducted regarding the effects by various combinations of TBs and PBs on three typical HiDS matrices generated by industrial applications currently in use.

The rest of this paper is organized as follows. Section 2 gives the preliminaries. Then Section 3 presents the LF model with different combinations of TBs and PBs. Section 4 provides and analyzes the experimental results. Finally, Section 5 concludes this paper.

Table 1
Different combinations of PBs in an LF model.

Situation	S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8
Combination Results a_{ui}	0	$\mu+b_u+b_i$	b_u+b_i	$\mu+b_u$	$\mu+b_i$	b_u	b_i	μ

2. Preliminaries

Given an item set I and a user set U, then user-item interactions can be denoted by a rating matrix $R^{|U|\times|I|}$ where each row denotes a specified user, each column denotes a specified item, and each entry r_{ui} indicating user u's preference on item i (note that r_{ui} may depend on either explicit or implicit preferences of user u on item i, and is usually proportional to those preferences). As discussed in the previous section, R is HiDS because a user can only access a small subset of I and vice versa. An LF model maps both users and items into a unique LF space of dimension f, modeming U and I with LF matrices $P^{|U|\times f}$ and $Q^{f\times|I|}$. Thus, each user u in U and item i in I are denoted by LF vectors p_u and q_i, and each entry r_{ui} in R is approximated by [19–21,23,24,26]:

$$\hat{r}_{ui} = p_u \cdot q_i, \tag{1}$$

where computes the inner product of two vectors.

Formula (1) approximates r_{ui} relying on the inner product of p_u and q_i, based on the assumption that the ratings information can be equally represented by a set of LFs [19–26]. However, in real applications, the characteristics of a recommender system are neither separately connected to its users nor items, e.g., the global-average PB μ; the user properties are hardly affected by the items, e.g., the user PB b_u; and the item properties are hardly affected by the users, e.g., the item PB b_i. Moreover, according to prior works in the area of ANN [29,30], it is also helpful to assume that principle component hides in desired LFs, which heavily affect the performance of resultant system. Hence, it is necessary to incorporate TBs as the additional components to the LF space. Therefore, the approximation (1) can be represented as the preprocessing bias [35,36] a_{ui} relying on PBs, training bias t_{ui} relying on TBs [31–34], and the inner product of p_u and q_i [19–24,26–30], resulting in:

$$\hat{r}_{ui} = a_{ui} + t_{ui} + p_u \cdot q_i. \tag{2}$$

3. LF model with TB and PB

3.1. Preprocessing biases

As discussed in prior works [19,23,32–34], an effective way to further improve the prediction accuracy of LF models for missing data in an HiDS matrix is incorporating the linear biases into the model. For a specified rating r_{ui} corresponding to user u and item i, the corresponding PB a_{ui} contain global-average μ, user PB b_u and item PB b_i, given by:

$$a_{ui} = \mu + b_u + b_i. \tag{3}$$

First of all, the global average μ represents the statistic characteristics of known data in R. Hence, it can be estimated by the average of the observed ratings [35] as follows:

$$\mu = \sum_{r_{ui}\in K} r_{ui}/|K|, \tag{4}$$

where K denotes the known entry set of R. Note that user and item PBs can be obtained through various designs of statistic estimators [22,23]. As demonstrated by previous works, the unbiased

Table 2
Computational rules of all combinations of PBs.

S.1	$\mu=0$ $b_i=0$ $b_u=0$
S.2	$\mu=\sum_{r_{ui}\in K} r_{ui}/\lvert K\rvert$ $b_i=\sum_{u\in R(i)}(r_{ui}-\mu)/(\theta_1+\lvert R(i)\rvert)$ $b_u=\sum_{i\in R(u)}(r_{ui}-\mu-b_i)/(\theta_2+\lvert R(u)\rvert)$
S.3	$\mu=0$ $b_i=\sum_{u\in R(i)} r_{ui}/(\theta_1+\lvert R(i)\rvert)$ $b_u=\sum_{i\in R(u)}(r_{ui}-b_i)/(\theta_2+\lvert R(u)\rvert)$
S.4	$\mu=\sum_{r_{ui}\in K} r_{ui}/\lvert K\rvert$ $b_i=0$ $b_u=\sum_{i\in R(u)}(r_{ui}-\mu)/(\theta_2+\lvert R(u)\rvert)$
S.5	$\mu=\sum_{r_{ui}\in K} r_{ui}/\lvert K\rvert$ $b_i=\sum_{u\in R(i)}(r_{ui}-\mu)/(\theta_1+\lvert R(i)\rvert)$ $b_u=0$
S.6	$\mu=0$ $b_i=0$ $b_u=\sum_{i\in R(u)} r_{ui}/(\theta_2+\lvert R(u)\rvert)$
S.7	$\mu=0$ $b_i=\sum_{u\in R(i)} r_{ui}/(\theta_1+\lvert R(i)\rvert)$ $b_u=0$
S.8	$\mu=\sum_{r_{ui}\in K} r_{ui}/\lvert K\rvert$ $b_i=0$ $b_u=0$

Table 3
Different combinations of TBs in an LF model.

Situation	S.A	S.B	S.C	S.D
Combination	c_u+c_i	0	c_u	c_i
Value of w_u	1	0	1	0
Value of w_i	1	0	0	1

linear estimators are able to boost the performance of the resultant model [22,23]. With this estimator, b_u and b_i are formulated by [22,23]:

$$b_i=\sum_{u\in R(i)}(r_{ui}-\mu)/(\theta_1+|R(i)|), \tag{5}$$

$$b_u=\sum_{i\in R(u)}(r_{ui}-\mu-b_i)/(\theta_2+|R(u)|), \tag{6}$$

where $R(i)$ denotes the set of users who have rated item i; $R(u)$ denotes the set of items that user u has rated; θ_1 and θ_2 represent the threshold constant related to the average of $R(i)$ and $R(u)$, respectively. Note that (5) depends on μ. Eq. (6) depends on μ and b_i. However, when considering different combinations of PBs, these formulas should be adjusted. In such cases, we can simply set the absent PBs at zero, making it not affect the final output. For instance, when we consider a PB combination consisting of b_i and b_u without μ by setting $\mu=0$, we arrive at:

$$\begin{aligned} b_i&=\sum_{u\in R(i)} r_{ui}/(\theta_1+|R(i)|),\\ b_u&=\sum_{i\in R(u)}(r_{ui}-b_i)/(\theta_2+|R(u)|). \end{aligned} \tag{7}$$

Based on such intuitions, we summarize the possible combinations of PBs in Table 1 and the calculation rules for PBs in different PB combinations in Table 2. Meanwhile, note that PBs are not trained along with LFs but relying on the prior estimators shown in Table 2. Hence, for incorporating them into an LF model, we can simply train the desired TBs and LFs on the residual matrix of R by removing the involved PBs from its each known entries:

$$r_{ui}^{res}=r_{ui}-a_{ui}, \tag{8}$$

where r_{ui}^{res} denotes the residual after removing PBs from r_{ui}. The formulation of a_{ui} depends on the adopted PB combinations summarized in Table 2.

3.2. Training bias

Based on the above inferences, PB combinations, i.e., S.1–S.8, are obtained through Tables 1 and 2. Because the PBs are not trained along with other LFs but estimated through the unbiased estimators, the desired LF matrices are actually constructed on the residuals of the observed ratings. With (8), PB combinations results in a residual set of κ, and each residual is approximated by corresponding LFs and TBs as follows:

$$r_{ui}^{res}=t_{ui}+p_u\cdot q_i, \tag{9}$$

As discussed in prior works [22–36], t_{ui} mainly consists of user and item TBs, leading to the following formulation:

$$t_{ui}=c_u+c_i, \tag{10}$$

where c_u and c_i denote the user and item TBs, respectively. According to (2), (8), (9) and (10), we formulate the objective function of an LF model with Euclidean distance as follows:

$$RSE=\sum_{r_{ui}\in K}\begin{pmatrix}(r_{ui}-a_{ui}-t_{ui}-p_u\cdot q_i)^2\\ +\lambda\left(\|p_u\|^2+\|q_i\|^2+w_uc_u^2+w_ic_i^2\right)\end{pmatrix}, \tag{11}$$

where $||\cdot||$ computes the l_2 norm of a vector; the term behind λ is the Tikhonov regularizing term for avoiding over-fitting the generalized error [31–35], and w_u and w_i denote the indicator weights

Table 4
Expressions of err_{ui} with different combinations of PBs and TBs.

Situation		PB							
		S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8
TB	S.A	r_{ui}-c_u-c_i-$p_u\cdot q_i$	r_{ui}-μ-b_u-b_i-c_u-c_i-$p_u\cdot q_i$	r_{ui}-b_u-b_i-c_u-c_i-$p_u\cdot q_i$	r_{ui}-μ-b_u-c_u-c_i-$p_u\cdot q_i$	r_{ui}-μ-b_i-c_u-c_i-$p_u\cdot q_i$	r_{ui}-b_u-c_u-c_i-$p_u\cdot q_i$	r_{ui}-b_i-c_u-c_i-$p_u\cdot q_i$	r_{ui}-μ-c_u-c_i-$p_u\cdot q_i$
	S.B	r_{ui}-$p_u\cdot q_i$	r_{ui}-μ-b_u-b_i-$p_u\cdot q_i$	r_{ui}-b_u-b_i-$p_u\cdot q_i$	r_{ui}-μ-b_u-$p_u\cdot q_i$	r_{ui}-μ-b_i-$p_u\cdot q_i$	r_{ui}-b_u-$p_u\cdot q_i$	r_{ui}-b_i-$p_u\cdot q_i$	r_{ui}-μ-$p_u\cdot q_i$
	S.C	r_{ui}-c_u-$p_u\cdot q_i$	r_{ui}-μ-b_u-b_i-c_u-$p_u\cdot q_i$	r_{ui}-b_u-b_i-c_u-$p_u\cdot q_i$	r_{ui}-μ-b_u-c_u-$p_u\cdot q_i$	r_{ui}-μ-b_i-c_u-$p_u\cdot q_i$	r_{ui}-b_u-c_u-$p_u\cdot q_i$	r_{ui}-b_i-c_u-$p_u\cdot q_i$	r_{ui}-μ-c_u-$p_u\cdot q_i$
	S.D	r_{ui}-c_i-$p_u\cdot q_i$	r_{ui}-μ-b_u-b_i-c_i-$p_u\cdot q_i$	r_{ui}-b_u-b_i-c_i-$p_u\cdot q_i$	r_{ui}-μ-b_u-c_i-$p_u\cdot q_i$	r_{ui}-μ-b_i-c_i-$p_u\cdot q_i$	r_{ui}-b_u-c_i-$p_u\cdot q_i$	r_{ui}-b_i-c_i-$p_u\cdot q_i$	r_{ui}-μ-c_i-$p_u\cdot q_i$

for c_u and c_i, respectively. Similar to PBs, for an LF model the best performance is not necessarily connected with the full appearance of both PBs. Hence, the value of wu and wi depend on the specified PB combinations adopted in an LF model, as depicted in Table 3.

Let err_{ui} denote the term r_{ui}-a_{ui}-t_{ui}-p_u•q_i. Based on the above inferences, we summarize all possible expressions of err_{ui} with different combinations of PB and TB in Table 4. Thus, we reformulate (11) as follows:

$$RSE = \sum_{r_{ui}\in K} \left({err_{ui}}^2 + \lambda\left(\|p_u\|^2 + \|q_i\|^2 + w_u c_u^2 + w_i c_i^2 \right) \right). \tag{12}$$

With (12), we clearly see that different combinations of PB and TB do not lead to significant change of parameter update rules. For instance, considering the case S.A-S1 in Table 4 where $w_u = w_i = 1$ as shown in Table 3. Then (12) is reformulated into:

$$RSE = \sum_{r_{ui}\in K} \left({err_{ui}}^2 + \lambda\left(\|p_u\|^2 + \|q_i\|^2 + c_u^2 + c_i^2 \right) \right), \tag{13}$$

Then by applying the stochastic gradient descent (SGD) to (13) we obtain:

$$\forall r_{ui} \in \kappa :$$

$$\begin{aligned}
c_u &\leftarrow c_u - \eta \frac{\partial RSE_{ui}}{\partial c_u} = c_u - \eta \cdot 2(-err_{ui} + \lambda c_u),\\
c_i &\leftarrow c_i - \eta \frac{\partial RSE_{ui}}{\partial c_i} = c_i - \eta \cdot 2(-err_{ui} + \lambda c_i),\\
p_u &\leftarrow p_u - \eta \frac{\partial RSE_{ui}}{\partial p_u} = p_u - \eta \cdot 2(-q_i \cdot err_{ui} + \lambda p_u),\\
q_i &\leftarrow q_i - \eta \frac{\partial RSE_{ui}}{\partial q_i} = q_i - \eta \cdot 2(-p_u \cdot err_{ui} + \lambda q_i).
\end{aligned} \tag{14}$$

However, considering the case S.C-S.1 in Table 4 where $w_u = 1$ and $w_i = 0$ as shown in Table 3. Then (12) is reformulated into:

$$RSE = \sum_{r_{ui}\in K} \left({err_{ui}}^2 + \lambda\left(\|p_u\|^2 + \|q_i\|^2 + c_u^2 + c_i^2 \right) \right). \tag{15}$$

By applying SGD to (15) we achieve the following update rule:

$$\forall r_{ui} \in \kappa :$$

$$\begin{aligned}
c_u &\leftarrow c_u - \eta \frac{\partial RSE_{ui}}{\partial c_u} == c_u - \eta \cdot 2(-err_{ui} + \lambda c_u),\\
p_u &\leftarrow p_u - \eta \frac{\partial RSE_{ui}}{\partial p_u} = p_u - \eta \cdot 2(-q_i \cdot err_{ui} + \lambda p_u),\\
q_i &\leftarrow q_i - \eta \frac{\partial RSE_{ui}}{\partial q_i} = q_i - \eta \cdot 2(-p_u \cdot err_{ui} + \lambda q_i).
\end{aligned} \tag{16}$$

By comparing (14) and (16), we see that they differ from each other in the following two aspects only:

(a) The expression of err_{ui}. As recorded in Table 4, with different combinations of PB and TB, the expression of err_{ui} varies; and
(b) Number of parameters to be updated. Note that PBs will not be involved in the parameter training process. However, different TB combinations will affect the number of parameters updated in each training iteration. On the other hand, as shown in (14) and (16), for involved parameters, their update rules are nearly the same with different TB combinations, except that the expression of err_{ui} changes.

Based on the above inferences, we summarize the general parameter update rule of LF models with different combinations of PB and TB as follows:

$$\forall r_{ui} \in \kappa :$$

$$\begin{aligned}
&if w_u = 1 : c_u \leftarrow c_u - \eta \cdot 2(-err_{ui} + \lambda c_u),\\
&if w_i = 1 : c_i \leftarrow c_i - \eta \cdot 2(-err_{ui} + \lambda c_i),\\
&p_u \leftarrow p_u - \eta \cdot 2(-q_i \cdot err_{ui} + \lambda p_u),\\
&q_i \leftarrow q_i - \eta \cdot 2(-p_u \cdot err_{ui} + \lambda q_i).
\end{aligned} \tag{17}$$

Note that the expression of err_{ui} depends on the adopted combination of PB and TB according to Table 4.

4. Experimental results and analysis

4.1. Evaluation protocol

For industrial applications, one major motivation to factorize a HiDS matrix is to estimate its missing entries. Owing to its popularity and significance, we adopt it as the evaluation protocol to validate involved models' performance. For a tested model, its prediction accuracy for missing entries in a HiDS matrix can be measured by root mean squared error (RMSE) [37] and mean absolute error (MAE):

$$RMSE = \sqrt{\left(\sum_{u,i\in T} \left(r_{ui} - \hat{r}_{ui} \right)^2 \right) / |T|}, \tag{18}$$

$$MAE = \left(\sum_{u,i\in T} \left| r_{ui} - \hat{r}_{ui} \right|_{abs} \right) / |T|, \tag{19}$$

where T denotes the testing set and $|\cdot|_{abs}$ denotes the absolute value of a given number. Naturally, we have $\kappa \bigcap T = \phi$ in the experiments. For an LF model, low RMSE and MAE denote high prediction accuracy. All experiments are conducted on a PC with a 3.4 GHz i7 CPU and 16 GB RAM.

4.2. Datasets

The experiments are conducted on three HiDS matrices, which are all collected by well-known and industrial applications of recommender systems [12,33]. The details of adopted datasets are listed in Table 5. Note that the known entries of each matrix scatter in the range of [0, 5].

4.3. Model settings

Note that our experiments are designed for checking how different LB combinations affect the performance of LF models. Hence, the hyper-parameters are set as $\eta = 0.01$ and $\lambda = 0.01$ uniformly for achieving objective results that are not affected by the hyper-parameter settings. Note that such hyper parameter settings are frequently adopted by prior works [19–25].

Meanwhile, to minimize the impact caused by differently and randomly initial hypotheses for desired LFs, tested LF models with different LB combinations are initialized with the same randomly-generated arrays (note that with different LF combinations, the initial hypotheses of two LF models cannot be exactly the same). Moreover, we have applied five-fold cross validation to each dataset to enhance the objectiveness of our experiments.

Table 5
Dataset in details.

No.	User count	Item count	Entry count	Description
D.1	138,493	26,744	20,000,263	MovieLens 20M matrix collected by the GroupLens reaserch group via the online recommender MovieLens.
D.2	755,720	120,492	13,668,320	Epinion dataset derived from Epinions.com.
D.3	129,490	58,541	16,830,839	Douban matrix collected form Douban.com.

Table 6
RMSE and MAE of different combinations on D.1.

D.1		RMSE								MAE							
		S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8	S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8
$f = 5$	S.A	0.8075	**0.8018**	0.8096	0.8036	0.8029	0.8019	0.8095	0.8019	0.6300	0.6244	0.6286	0.6250	**0.6241**	0.6245	0.6285	0.6250
	S.B	0.8234	**0.8067**	0.8142	0.8070	0.8069	0.8070	0.8217	0.8074	0.6424	**0.6250**	0.6333	0.6271	0.6283	0.6277	0.6377	0.6282
	S.C	0.8178	**0.8040**	0.8188	0.8048	0.8041	0.8043	0.8177	0.8043	0.6364	**0.6249**	0.6365	0.6257	0.6252	0.6255	0.6344	0.6258
	S.D	0.8173	**0.8034**	0.8081	0.8056	0.8052	0.8053	0.8101	0.8053	0.6366	**0.6248**	0.6279	0.6266	0.6270	0.6278	0.6297	0.6275
$f = 10$	S.A	0.7918	**0.7872**	0.7981	0.7875	0.7873	0.7878	0.7984	0.7880	0.6170	**0.6114**	0.6178	0.6121	0.6115	0.6115	0.6180	0.6119
	S.B	0.8090	0.7899	0.7969	0.7906	**0.7894**	0.7896	0.8001	0.7897	0.6298	0.6128	0.6187	0.6127	**0.6127**	0.6133	0.6195	0.6133
	S.C	0.8039	0.7888	0.8005	0.7894	**0.7887**	0.7891	0.8004	0.7890	0.6260	0.6124	0.6200	0.6132	**0.6122**	0.6127	0.6197	0.6128
	S.D	0.7973	**0.7879**	0.7911	0.7882	0.7880	0.7884	0.7942	0.7880	0.6211	**0.6111**	0.6133	0.6119	0.6128	0.6124	0.6157	0.6121
$f = 20$	S.A	0.7817	0.7785	0.7891	0.7784	**0.7780**	0.7781	0.7893	0.7781	0.6075	0.6036	0.6101	0.6040	**0.6034**	0.6037	0.6105	0.6039
	S.B	0.7988	0.7791	0.7870	0.7795	**0.7781**	0.7797	0.7879	0.7791	0.6212	0.6034	0.6087	0.6036	**0.6033**	0.6046	0.6092	0.6044
	S.C	0.7938	0.7789	0.7911	0.7806	**0.7785**	0.7796	0.7908	0.7799	0.6174	0.6037	0.6112	0.6049	**0.6036**	0.6048	0.6110	0.6049
	S.D	0.7854	0.7779	0.7810	0.7778	**0.7772**	0.7785	0.7834	0.7771	0.6098	0.6030	0.6051	0.6032	**0.6027**	0.6038	0.6063	0.6032
$f = 40$	S.A	0.7767	0.7730	0.7834	0.7734	**0.7727**	0.7729	0.7840	0.7728	0.6025	0.5990	0.6056	0.5994	**0.5989**	0.5992	0.6060	0.5995
	S.B	0.7765	0.7729	0.7839	0.7735	**0.7728**	0.7729	0.7841	0.7732	0.6177	0.5981	0.6035	0.5989	**0.5981**	0.5998	0.6040	0.5996
	S.C	0.7764	0.7734	0.7835	0.7734	**0.7727**	0.7728	0.7841	0.7726	0.6135	0.5989	0.6064	0.6008	**0.5987**	0.6001	0.6062	0.6006
	S.D	0.7765	0.7730	0.7839	0.7735	**0.7728**	0.7729	0.7841	0.7732	0.6046	0.5984	0.5998	0.5978	**0.5983**	0.5990	0.6018	0.5983

Table 7
Training round count of different combinations on D.1.

D.1		Training round							
		S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8
$f = 5$	S.A	40	**20**	112	21	20	20	112	20
	S.B	60	**15**	40	25	21	31	37	26
	S.C	339	**22**	32	25	24	23	31	23
	S.D	30	**17**	24	22	22	22	39	21
$f = 10$	S.A	20	**17**	31	18	17	18	37	17
	S.B	37	20	24	20	**18**	21	25	19
	S.C	30	18	24	19	**18**	19	24	19
	S.D	19	**17**	20	18	18	19	23	18
$f = 20$	S.A	17	17	20	16	**16**	17	21	17
	S.B	26	17	18	17	**16**	17	18	17
	S.C	21	17	19	17	**17**	17	19	17
	S.D	17	16	17	17	**16**	17	18	17
$f = 40$	S.A	16	16	17	16	**16**	16	17	16
	S.B	16	16	17	16	**16**	16	17	16
	S.C	16	16	17	16	**16**	16	17	16
	S.D	16	16	17	16	**16**	16	17	16

Table 8
RMSE and MAE of different combinations on D.2.

D.2		RMSE								MAE							
		S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8	S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8
$f = 5$	S.A	0.5256	0.4680	0.5432	0.4686	**0.4606**	0.5319	0.5310	0.4609	0.2992	0.2742	0.2931	0.2744	**0.2717**	0.2927	0.2899	0.2718
	S.B	0.6059	0.4803	0.6145	0.4866	**0.4683**	0.5761	0.6113	0.4740	0.3146	0.2829	0.3310	0.2692	**0.2684**	0.3036	0.3184	0.2707
	S.C	0.5821	0.4813	0.6085	0.4871	**0.4723**	0.5629	0.5830	0.4771	0.3107	0.2798	0.3227	0.2657	**0.2763**	0.2893	0.3148	0.2635
	S.D	0.5483	0.4707	0.5536	0.4728	**0.4575**	0.5496	0.5367	0.4570	0.3009	0.2800	0.3002	0.2767	**0.2629**	0.3022	0.2819	0.2630
$f = 10$	S.A	0.5272	0.4691	0.5446	0.4695	**0.4613**	0.5330	0.5319	0.4614	0.2993	0.2743	0.2945	0.2737	**0.2707**	0.2925	0.2893	0.2711
	S.B	0.6065	0.4811	0.6173	0.4874	**0.4688**	0.5756	0.6133	0.4747	0.3163	0.2823	0.3317	0.2690	**0.2676**	0.3057	0.3224	0.2706
	S.C	0.5833	0.4812	0.6092	0.4887	**0.4724**	0.5635	0.5838	0.4763	0.3120	0.2792	0.3249	0.2644	**0.2631**	0.2900	0.3134	0.2753
	S.D	0.5493	0.4711	0.5548	0.4735	**0.4576**	0.5500	0.5386	0.4577	0.3009	0.2784	0.2995	0.2759	**0.2623**	0.3018	0.2818	0.2631
$f = 20$	S.A	0.5300	0.4704	0.5467	0.4716	**0.4625**	0.5350	0.5337	0.4625	0.2997	0.2738	0.2935	0.2733	**0.2701**	0.2919	0.2888	0.2704
	S.B	0.6078	0.4833	0.6187	0.4888	**0.4698**	0.5772	0.6142	0.4768	0.3165	0.2814	0.3318	0.2680	**0.2670**	0.3053	0.3223	0.2706
	S.C	0.5846	0.4836	0.6106	0.4892	**0.4735**	0.5662	0.5846	0.4782	0.3119	0.2788	0.3246	0.2642	**0.2627**	0.2902	0.3127	0.2640
	S.D	0.5513	0.4729	0.5568	0.4752	0.4595	0.5515	0.5398	**0.4590**	0.3007	0.2783	0.3003	0.2756	0.2621	0.3015	0.2817	**0.2620**
$f = 40$	S.A	0.5327	0.4726	0.5487	0.4728	**0.4627**	0.5374	0.5358	0.4628	0.3004	0.2739	0.2937	0.2736	**0.2707**	0.2922	0.2890	0.2708
	S.B	0.6095	0.4835	0.6195	0.4918	**0.4717**	0.5783	0.6139	0.4795	0.3168	0.2816	0.3317	0.2681	**0.2675**	0.3054	0.3226	0.2711
	S.C	0.5866	0.4837	0.6099	0.4917	**0.4744**	0.5674	0.5855	0.4798	0.3118	0.2789	0.3230	0.2645	**0.2629**	0.2901	0.3122	0.2643
	S.D	0.5537	0.4751	0.5590	0.4765	**0.4605**	0.5533	0.5421	0.4606	0.3011	0.2784	0.3002	0.2757	**0.2622**	0.3014	0.2821	0.2625

Table 9
Training round count of different combinations on D.2.

D.2		Training round							
		S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8
$f=5$	S.A	62	48	68	48	**43**	53	57	43
	S.B	62	51	103	56	**45**	71	93	48
	S.C	67	50	91	55	**45**	72	74	50
	S.D	58	50	68	47	**43**	58	62	44
$f=10$	S.A	58	47	63	47	**40**	51	55	42
	S.B	57	48	87	53	**44**	62	78	46
	S.C	61	49	78	51	**44**	64	66	48
	S.D	55	49	62	46	**43**	54	58	43
$f=20$	S.A	56	45	61	43	**35**	50	53	38
	S.B	56	**24**	81	51	43	60	75	45
	S.C	60	**23**	72	51	41	61	63	46
	S.D	53	47	60	44	**40**	52	57	41
$f=40$	S.A	55	37	61	28	**26**	49	53	27
	S.B	56	**23**	84	25	39	59	78	40
	S.C	61	**23**	71	26	34	60	63	41
	S.D	52	44	60	30	**27**	50	57	27

4.4. Results

Tables 6, 8 and 10 record the RMSE and MAE of LF models with different LB combinations on all three datasets and the LF dimension f increases from 5 to 40. Tables 7, 9 and 11 record the converging iteration count of each LB combination. Figs. 1–3 depict the typical training process of LF models with different LB combinations, respectively. According to these results, we have the following findings:

(a) On D.1, LF models with PB combinations S.2 and S.5 have better performance than LF models with the other PB combinations when $f=5$ and $f=10$. Moreover, as f increases over 20, S.5 has the highest prediction accuracy. The lowest RMSE is 0.8018, 0.7872, 0.7772, 0.7727 when the LF dimension set to $f=5$, $f=10$, $f=20$, $f=40$, respectively. Similar situations can be found on D.3: LF models with PB combinations S.2 and S.5 take turns to achieve the lowest prediction error.

On D2, the situation is a bit different since the LF model with the PB combination S.5 always perform better than its peers except for the testing case with $f=20$ and TB combination S.D. Nonetheless, in that case the RMSE and MAE of the LF model with the PB combination S.5 is still very close to the winner, i.e., the LF model with the PB combination S.8: the RMSE is 0.4595 versus 0.4590, and the MAE is 0.2621 versus 0.2620.

From Table 2, we see that both S.2 and S5 contain the PBs μ and b_i. From such results, we assume that the global average μ and the item average b_i can better describe the prior knowledge hidden in κ. On the other hand, we see that the user average b_u may lead to the opposite results, i.e., the accuracy of an LF recommender may decrease as shown in Tables 6, 8 and 10 with b_u. Such phenomena maybe interpreted as follows: the user taste is far less stable than the global trend and item evaluation, making the user average b_u not so informative as the global and item average.

(b) According to Tables 7, 9 and 11, we see that an LF models' converging iteration count and prediction accuracy for missing data in an HiDS matrix are closely related with its adopted LB combinations. It is exciting to see that with the optimal LB combination, an LF model is able to achieve high prediction accuracy and fast convergence simultaneously in most cases. However, exceptions are occasionally encountered. For instance, for the LF model with the LB combination S.D/S.8 with $f=20$ on D.2, its RMSE is 0.4590 which is the lowest, but it converges with 41 iterations which is not the least among its peers, as recorded in Tables 8 and 9.

Moreover, from Figs. 1 to 3 we clearly see that the training curve of an LF model varies with different LB combinations. The start points of RMSE is different on three datasets when we incorporate LB combinations into LF model as depicted in Figs. 1–3. In general, an LF model starts with lower training error with the integration of more LBs (either PB or TB). The reason for this phenomenon is two-fold:

With the integration of more PBs, more prior information contains in the statistics of κ is passed to the LF model at the very beginning, which improves the quality of internally generated predictions by the insufficiently trained model; and

(2) TBs actually play the role of principle LFs according to (11): they are linearly combined with the product of LFs and can heavily affect the outcome prediction of an LF model, especially at the beginning of its training process. Hence, more TBs will also result in low prediction error of an LF model at the beginning of its training process.

However, as depicted in Figs. 1–3, low prediction error at the very beginning of the training process of an LF model does not ensure that it is able to achieve the best local optima (although in most cases it appears so but exceptions are occasionally encountered, as given in Figs. 1–3). Reason for this phenomenon is still unveiled and calls for our future efforts.

(c) Considering the TB combinations, they perform differently as the PB combination varies. For example, when $f=20$ on D1, the RMSE of LF models with the PB combination S.1 and all four TB combinations S.A–S.D is 0.7817, 0.7988, 0.7938 and 0,7854, where S.A makes the LF model to achieve the lowest RMSE. Nonetheless, when $f=20$ on D1, the RMSE of LF models with the PB combination S.3 and all four TB combinations S.A–S.D is 0.7891, 0.7870, 0.7911 and 0.7810, where S.D makes the LF model to achieve the lowest RMSE.

(d) Based on the experimental results, we draw significance tests to validate that which LB combinations are able to achieve significant advantage in prediction accuracy. We choose to perform the Friedman test [38], owing to its effectiveness in validating the performance of multiple models on multiple datasets. Let r_i^j be the rank of the tested models on corresponding testing cases, where j denotes the jth one of k tested models and i denotes the ith one of N testing cases, respectively. The Friedman test compares each average rank of involved models. Under the null-hypothesis that all the algorithms are equivalent, their ranks R_j should be equal. Thus, the Friedman value is given by:

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right]. \tag{20}$$

According to (20), the test score is given by

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2}. \tag{21}$$

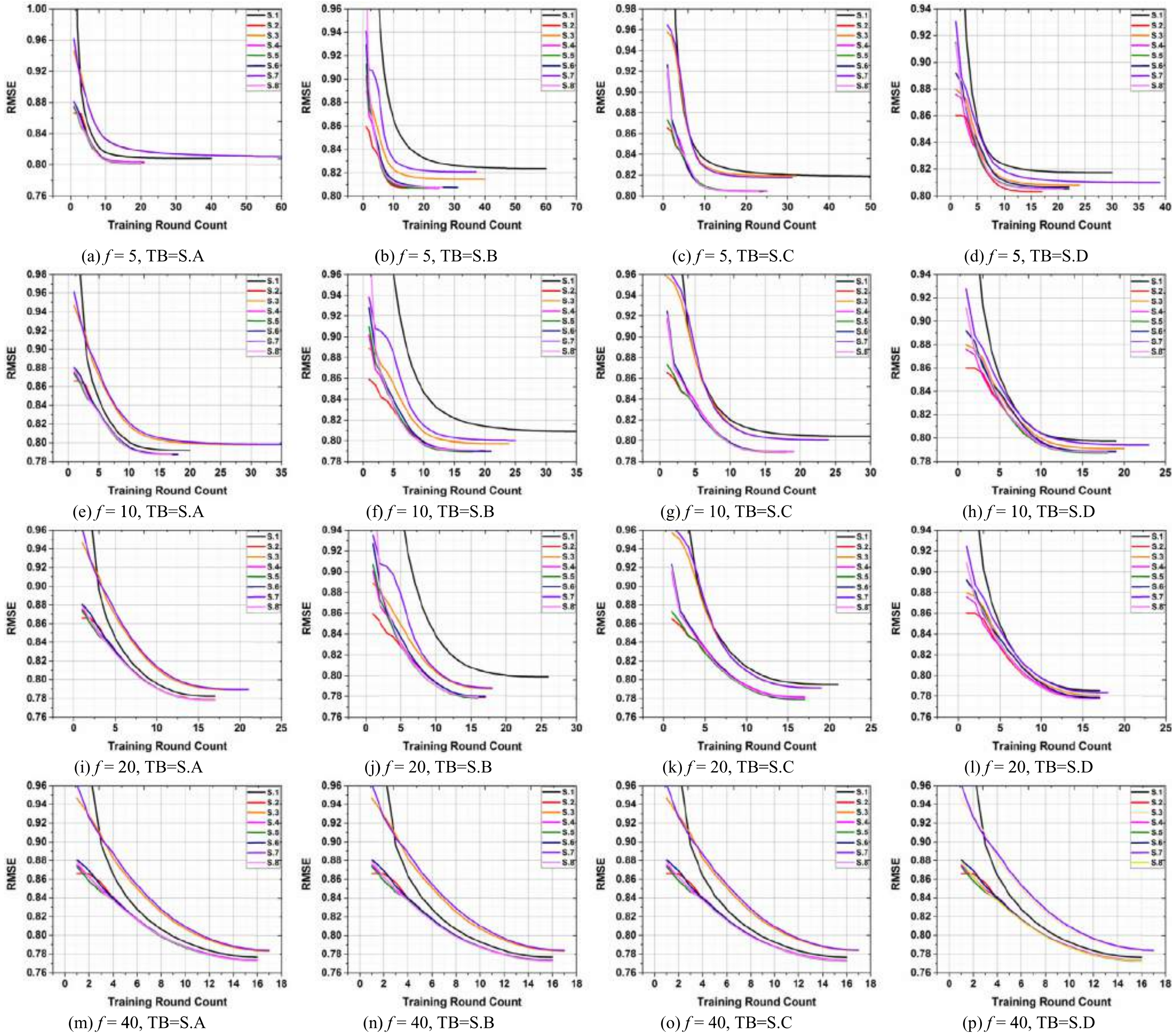
Note that (21) is distributed according to the F-distribution with $k-1$ and $(k-1)$ $(N-1)$ degrees of freedom [38]. Hence, we can reject the null hypothesis with the critical level α, if F_F is greater than the corresponding critical value.

In our experiments, 32 models are tested on 24 testing cases when considering the prediction accuracy of an LF model in both RMSE and MAE. Based on the records from Tables 6, 8 and 10, we compute the average rank of each tested model in prediction accuracy on all datasets in Table 12.

Thus, the k and N in (20) and (21) are 32 and 24 respectively, and F_F in our case is distributed according to the F-distribution with $32-1=31$ and $(32-1)$ $(24-1)=713$ degrees of freedom. The critical value of $F(31, 713)$ for $\alpha=0.05$ is 1.41. Therefore, if the testing scores of our experiments are greater than it, we reject the

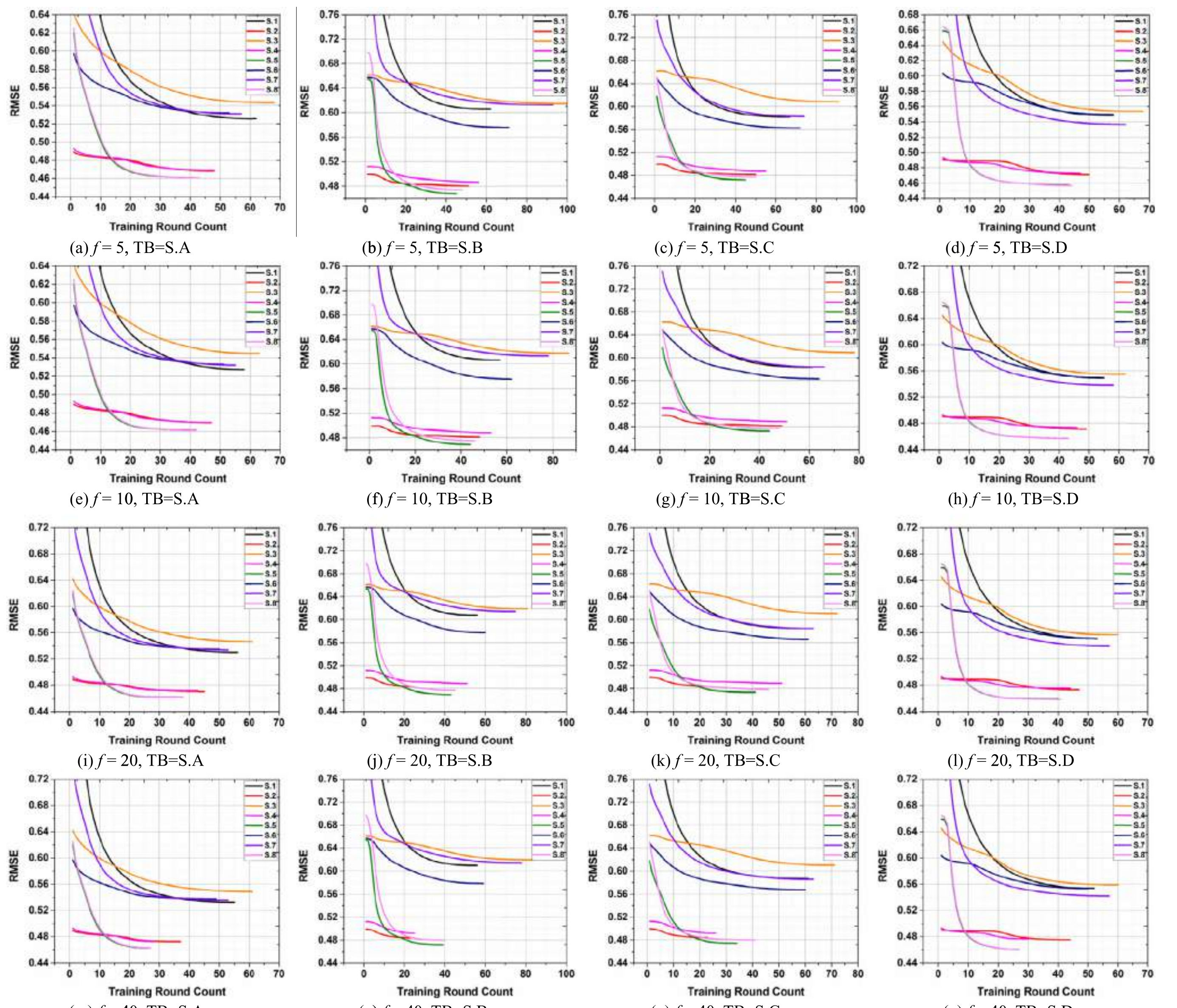
Table 10
RMSE and MAE of different combinations on D.3.

D.3		RMSE								MAE							
		S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8	S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8
$f = 5$	S.A	0.7163	0.7058	0.7193	0.7056	**0.7052**	0.7124	0.7203	0.7057	0.5699	0.5631	0.5696	0.5632	**0.5628**	0.5655	0.5696	0.5633
	S.B	0.7373	**0.7041**	0.7259	0.7061	0.7051	0.7167	0.7302	0.7068	0.5819	**0.5615**	0.5731	0.5630	0.5621	0.5679	0.5753	0.5637
	S.C	0.7270	**0.7054**	0.7240	0.7069	0.7052	0.7129	0.7247	0.7062	0.5777	**0.5615**	0.5717	0.5629	0.5617	0.5653	0.5714	0.5636
	S.D	0.7305	**0.7045**	0.7175	0.7060	0.7057	0.7163	0.7229	0.7058	0.5769	**0.5626**	0.5687	0.5641	0.5639	0.5670	0.5701	0.5641
$f = 10$	S.A	0.7150	**0.7024**	0.7180	0.7027	0.7025	0.7093	0.7183	0.7027	0.5672	**0.5596**	0.5670	0.5600	0.5601	0.5630	0.5678	0.5603
	S.B	0.7357	**0.6999**	0.7242	0.7020	0.7006	0.7150	0.7303	0.7022	0.5788	**0.5578**	0.5701	0.5595	0.5583	0.5647	0.5721	0.5603
	S.C	0.7261	0.7017	0.7218	0.7037	**0.7010**	0.7110	0.7221	0.7029	0.5748	0.5581	0.5693	0.5598	**0.5579**	0.5629	0.5688	0.5598
	S.D	0.7273	**0.7013**	0.7154	0.7019	0.7018	0.7127	0.7213	0.7019	0.5725	**0.5591**	0.5652	0.5601	0.5602	0.5642	0.5671	0.5600
$f = 20$	S.A	0.7126	**0.6996**	0.7157	0.6999	0.6998	0.7073	0.7172	0.6999	0.5656	**0.5573**	0.5651	0.5580	0.5575	0.5608	0.5660	0.5579
	S.B	0.7363	**0.6978**	0.7238	0.6995	0.6977	0.7131	0.7333	0.6999	0.5782	**0.5551**	0.5687	0.5569	0.5554	0.5627	0.5718	0.5574
	S.C	0.7253	0.6987	0.7202	0.7007	**0.6983**	0.7093	0.7209	0.7005	0.5744	0.5554	0.5669	0.5578	**0.5554**	0.5607	0.5672	0.5574
	S.D	0.7245	**0.6985**	0.7133	0.6991	0.6990	0.7101	0.7193	0.6989	0.5698	**0.5565**	0.5631	0.5574	0.5573	0.5620	0.5655	0.5576
$f = 40$	S.A	0.7111	0.6978	0.7136	0.6980	**0.6976**	0.7054	0.7155	0.6977	0.5636	0.5558	0.5634	0.5561	**0.5555**	0.5589	0.5641	0.5560
	S.B	0.7355	0.6959	0.7224	0.6976	**0.6958**	0.7115	0.7329	0.6977	0.5773	0.5536	0.5671	0.5553	**0.5532**	0.5613	0.5707	0.5558
	S.C	0.7239	0.6965	0.7185	0.6990	**0.6964**	0.7073	0.7190	0.6986	0.5730	0.5536	0.5651	0.5561	**0.5535**	0.5591	0.5654	0.5558
	S.D	0.7223	**0.6965**	0.7115	0.6971	0.6969	0.7085	0.7176	0.6970	0.5678	**0.5545**	0.5613	0.5555	0.5554	0.5603	0.5634	0.5557



(a) $f = 5$, TB=S.A (b) $f = 5$, TB=S.B (c) $f = 5$, TB=S.C (d) $f = 5$, TB=S.D
(e) $f = 10$, TB=S.A (f) $f = 10$, TB=S.B (g) $f = 10$, TB=S.C (h) $f = 10$, TB=S.D
(i) $f = 20$, TB=S.A (j) $f = 20$, TB=S.B (k) $f = 20$, TB=S.C (l) $f = 20$, TB=S.D
(m) $f = 40$, TB=S.A (n) $f = 40$, TB=S.B (o) $f = 40$, TB=S.C (p) $f = 40$, TB=S.D

Fig. 1. Training process in RMSE of D.1.



(a) $f = 5$, TB=S.A (b) $f = 5$, TB=S.B (c) $f = 5$, TB=S.C (d) $f = 5$, TB=S.D
(e) $f = 10$, TB=S.A (f) $f = 10$, TB=S.B (g) $f = 10$, TB=S.C (h) $f = 10$, TB=S.D
(i) $f = 20$, TB=S.A (j) $f = 20$, TB=S.B (k) $f = 20$, TB=S.C (l) $f = 20$, TB=S.D
(m) $f = 40$, TB=S.A (n) $f = 40$, TB=S.B (o) $f = 40$, TB=S.C (p) $f = 40$, TB=S.D

Fig. 2. Training process in RMSE of D.2.

Table 11
Training round count of different combinations on D.3.

D.3		Training round							
		S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8
$f = 5$	S.A	41	22	39	21	**21**	29	44	22
	S.B	69	**19**	87	22	22	50	82	26
	S.C	51	**20**	41	24	22	35	42	25
	S.D	43	**19**	37	22	22	38	51	23
$f = 10$	S.A	25	**19**	25	20	20	21	26	19
	S.B	38	**19**	37	20	20	25	44	20
	S.C	29	20	27	21	**19**	23	27	21
	S.D	24	**19**	23	19	19	22	26	19
$f = 20$	S.A	20	**19**	20	19	19	19	21	19
	S.B	24	**18**	22	19	18	19	26	19
	S.C	21	19	20	19	**19**	19	20	19
	S.D	19	**19**	19	18	18	19	20	18
$f = 40$	S.A	19	18	19	18	**18**	18	19	18
	S.B	20	18	20	18	**18**	18	20	18
	S.C	19	18	19	18	**18**	19	19	18
	S.D	18	**18**	19	18	18	18	19	18

Table 12
Average ranks of all tested models in prediction accuracy.

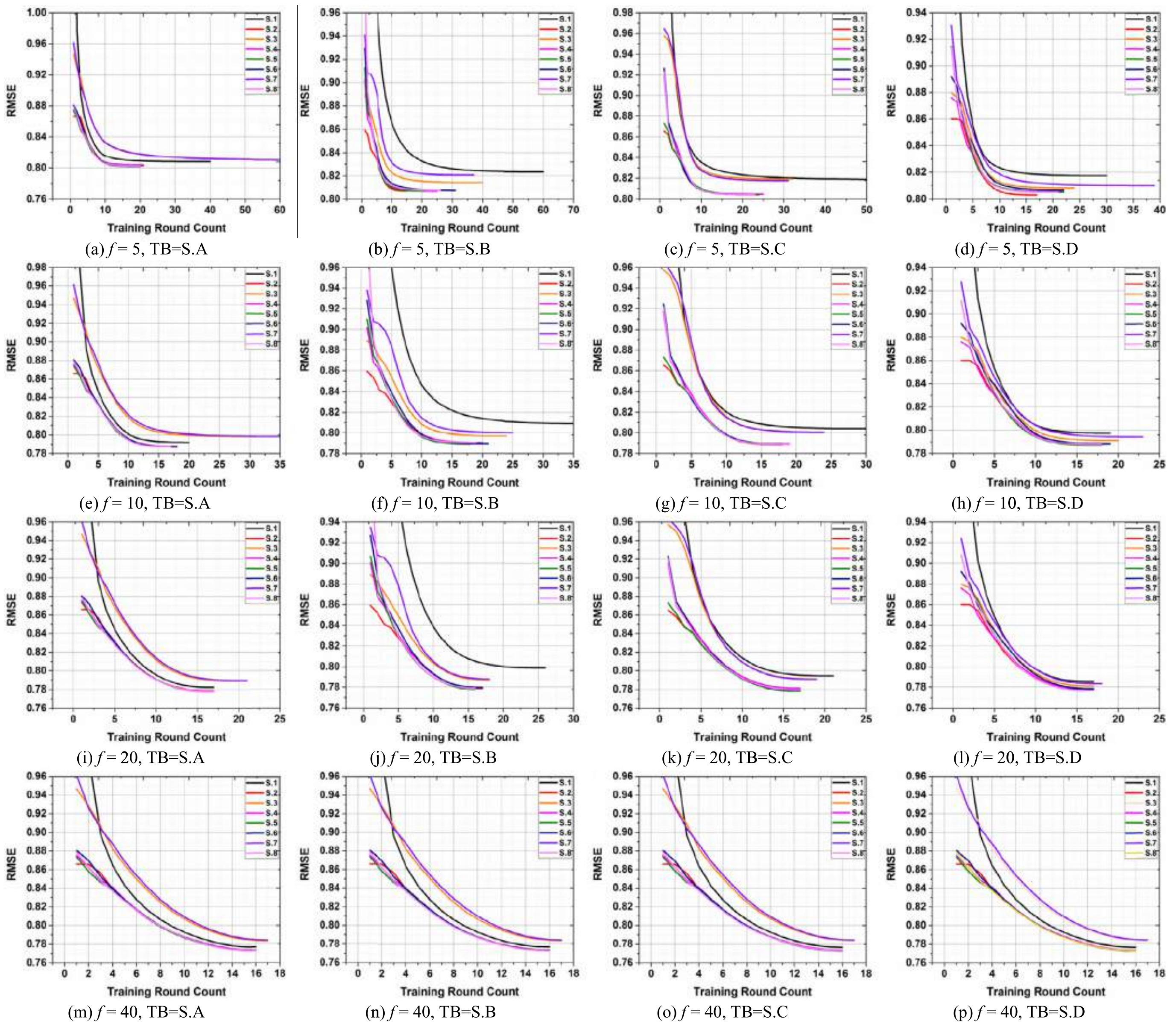
AvgRank	S.1	S.2	S.3	S.4	S.5	S.6	S.7	S.8
S.A	21.25	7.13	22.75	9.75	**6.83**	14.16	22.54	8.42
S.B	30.96	8.50	28.17	10.79	**6.25**	21.25	29.08	12.88
S.C	29.17	9.00	28.63	13.17	**7.63**	18.04	27.54	11.46
S.D	27.00	7.25	22.46	10.96	**6.33**	18.92	22.21	7.54

null hypothesis. Then, we compute the test scores as follows:

$$\chi^2 = \frac{12\times24}{32\times33}\left[\sum_j R_j^2 - \frac{32\times33^2}{4}\right] \tag{22}$$
$$= 101.14 \Rightarrow F_F = 3.62.$$

From (22) we find that all test scores are far greater than 1.41. Thus, we conclude that the tested models are significantly different in performance with a confidence of 95%.

In order to further identify the performance of tested models, we employ the Nemenyi analysis [30]. In the test, two models are significantly different if the difference value between their perfor-



(a) $f = 5$, TB=S.A (b) $f = 5$, TB=S.B (c) $f = 5$, TB=S.C (d) $f = 5$, TB=S.D

(e) $f = 10$, TB=S.A (f) $f = 10$, TB=S.B (g) $f = 10$, TB=S.C (h) $f = 10$, TB=S.D

(i) $f = 20$, TB=S.A (j) $f = 20$, TB=S.B (k) $f = 20$, TB=S.C (l) $f = 20$, TB=S.D

(m) $f = 40$, TB=S.A (n) $f = 40$, TB=S.B (o) $f = 40$, TB=S.C (p) $f = 40$, TB=S.D

Fig. 3. Training process in RMSE of D.3.

mance ranks is greater than the critical difference value, which is given by

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}, \tag{23}$$

where q_α is based on the Studentized range statistic [38]. With 32 models in the experiment, in our case the critical value $q_\alpha = 3.78$ with the critical level $\alpha = 0.1$ [30]. By substituting $k = 32$ and $N = 24$, we obtain $CD = 10.24$, which indicates that any pair of models with a rank difference higher than 10.24 have significant difference in prediction accuracy with a confidence of 90%.

The results of Nemenyi analysis based on the experimental results are depicted in Fig. 4. Firstly, we find that S.5 outperforms other models in terms of prediction accuracy. Meanwhile, S.2 is able to achieve comparable prediction accuracy with S.5. We can find that statistical overall average is always efficient for prediction accuracy. Meanwhile, item bias plays a positive role to an LF model while user bias even lead to a little negative effect from S.2 and S.5. Moreover, S.2 and S.5 are able to achieve significant improvement in prediction accuracy when compared with S.1, S.3 and S.7. However, this inference is not valuable for the other models.

Note that the case S.A-S.1 actually denotes an LF model with commonly adopted LB combination in prior works [19,23], and the case S.A-S.2 adopt all the biases simultaneously [32–35]. Fig. 4 and Table 12 indicate that the LB combinations S.A-S.1 and S.A-S.2 cannot lead to an LF model with the best performance. The average rank of S.A-S.1 and S.A-S.2 are 21.25 and 7.13, respectively. As shown in Fig. 4, the LB combination S.B-S.5 leads to the LF model with the highest average rank, i.e., 6.25. Moreover, we see that each PB/TB does have positive/negative effects in the performance of an LF model. For instance, the LB combination S.A-S.2 includes all the TBs and PBs but S.B-S.2 only includes PBs. From Fig. 4 we see that S.A-S.2 has a better performance than S.B-S.2 on statistics prediction accuracy. It indicates that TBs have positive effects in our experiments. So we can choose the optimum LB combination for different HiDS matrices.

(e) Figs. 5 and 6 depict the LF distributions of an LF model with PB combinations S.1–S.8, TB combination S.A and S.B, $f = 5$

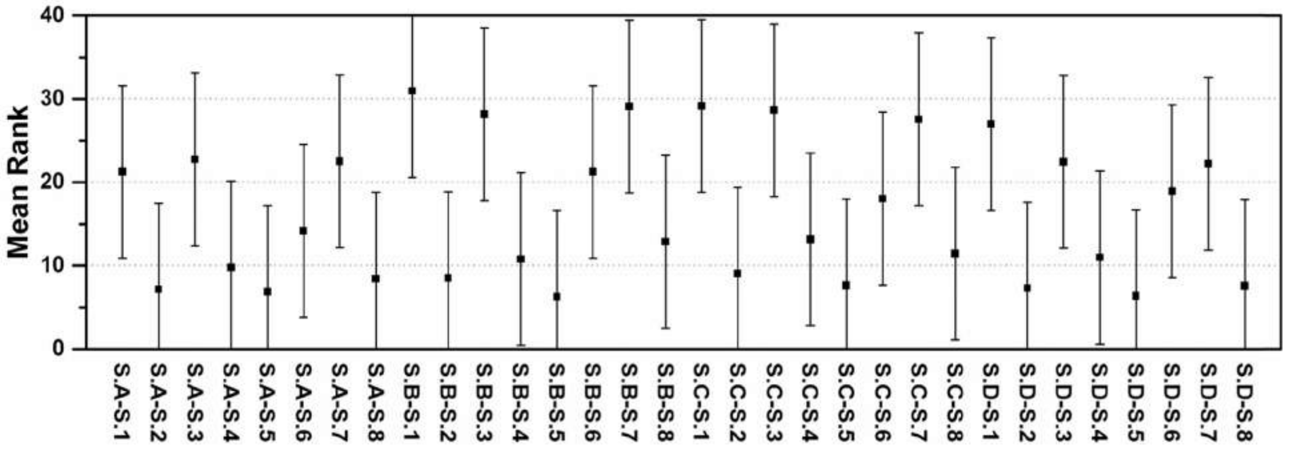
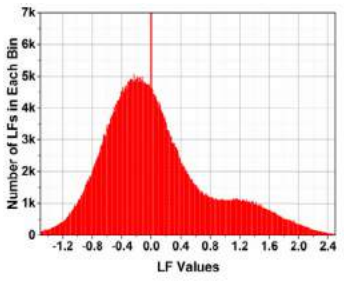
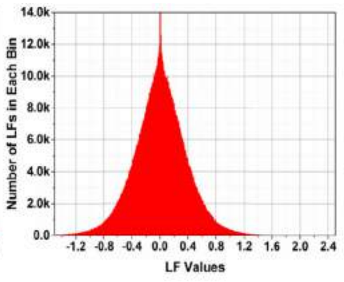
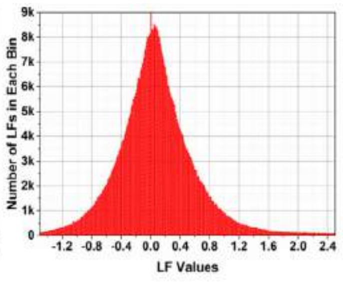
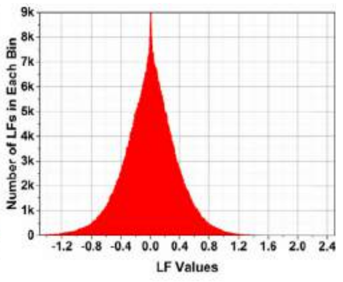


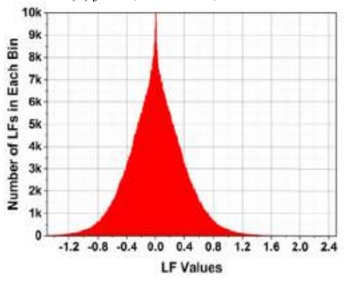
Fig. 4. Results of Nemenyi analysis.

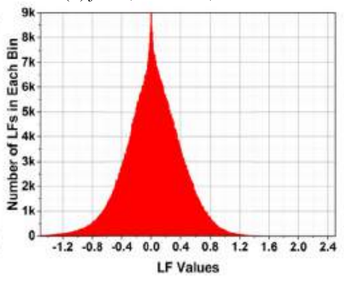

(a) *f* = 5, TB=S.A, PB=S.1

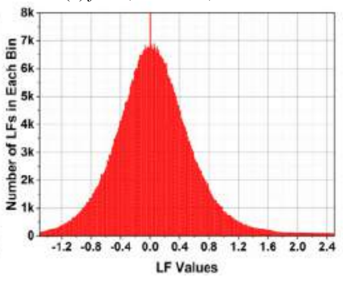

(b) *f* = 5, TB=S.A, PB=S.2


(c) *f* = 5, TB=S.A, PB=S.3


(d) *f* = 5, TB=S.A, PB=S.4


(e) *f* = 5, TB=S.A, PB=S.5


(f) *f* = 5, TB=S.A, PB=S.6


(g) *f* = 5, TB=S.A, PB=S.7

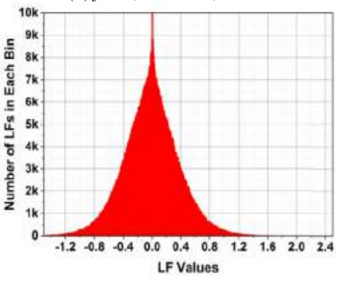
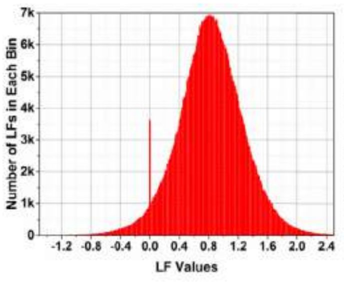
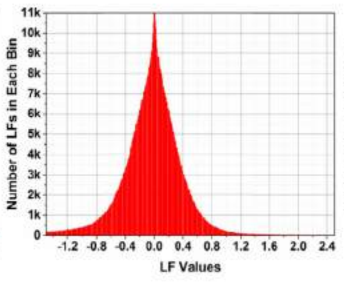
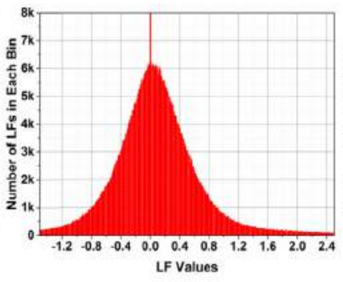
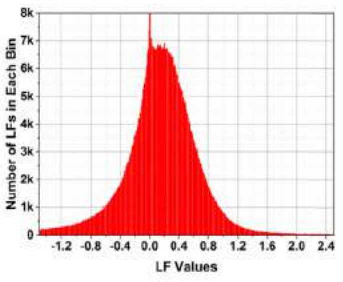

(h) *f* = 5, TB=S.A, PB=S.8

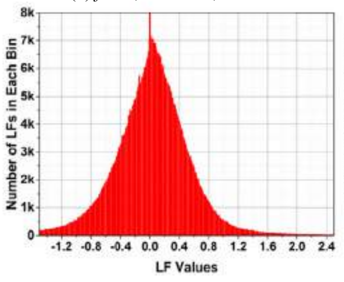
Fig. 5. Distributions of LFs in different biases on D.1 with TB = S.A.

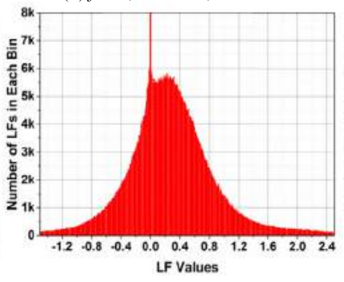

(a) *f* = 5, TB=S.B, PB=S.1

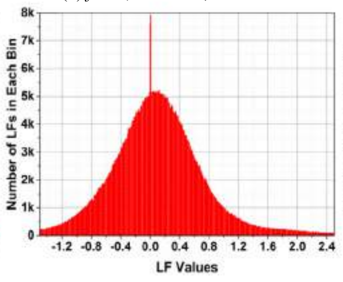

(b) *f* = 5, TB=S.B, PB=S.2


(c) *f* = 5, TB=S.B, PB=S.3


(d) *f* = 5, TB=S.B, PB=S.4


(e) *f* = 5, TB=S.B, PB=S.5


(f) *f* = 5, TB=S.B, PB=S.6


(g) *f* = 5, TB=S.B, PB=S.7

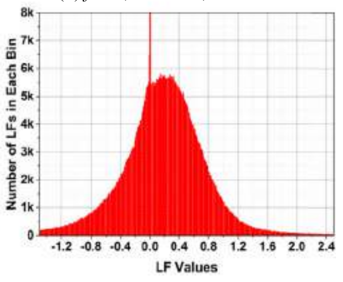

(h) *f* = 5, TB=S.B, PB=S.8

Fig. 6. Distributions of LFs in different biases on D.1 with TB = S.B.

on D.1. However, similar situations can be found with TB combinations S.C–S.D and $f = 10$, 20, and 40 on all datasets. From Figs. 5 and 6, we find that the LF distributions mainly depend on the PBs instead of TBs in our experiments. Moreover, LF distributions tend to concentrate around a certain point with the incorporation of PBs. For instance, without any PBs, the LF distribution is obviously asymmetric as shown in Figs. 5(a) and 6(a). Nonetheless, the symmetry of LF distributions becomes obvious after the integration of PBs into an LF model, as shown in Figs. 5(b)–(h) and 6(b)–(h).

(f) *Summary:* Based on the detailed empirical studies, we conclude that: (a) the global average μ is the most important PB which always brings stable performance gain into an LF model, while user PB b_u and item PB b_i have data-dependently positive/negative effects; (b) Incorporating TBs into an LF model is beneficial to accelerate the convergence rate of an LF model; (c) in most testing cases, the optimal LB combination can implement the highest prediction accuracy and convergence rate simultaneously; (d) the effects of LBs decreases as the LF dimension increases; (g) the integration of PBs into an LF model results in symmetric and concentrative LF distributions.

5. Conclusions

In this work, we aim at validating the effects of LBs, including the preprocessing biases (PBs) and training biases (TBs) [31–35] in the performance of an LF model. To do so, we first analyze the integration strategy of PBs and TBs into an LF model, along with the possible combinations of PBs and TBs. Subsequently, we derive different parameter update rules corresponding to different PB and TB combinations in an LF model relying on Euclidean distance as the loss function and stochastic gradient descent as the training scheme. Afterwards, detailed empirical studies are conducted on three high-dimensional and sparse matrices generated by industrial applications currently in use, for identifying different effects by different LB combinations. The experimental results demonstrate that different LB combinations do have different effects in the performance of an LF model, while several of them proves to be important in constantly bringing positive effects into the prediction accuracy and convergence rate of an LF model.

This paper is based on empirical studies without considering the non-negativity constraints in an LF model. However, according to prior works, non-negativity is a vital characteristic of most industrial data [26,33–36]. Hence, it is necessary to identify their effects of LBs in non-negative LF models and theoretically prove these results. Meanwhile, it is also desired to apply the LB strategies to the LF analysis on other industrial applications like cloud computing [39–40], protein interactome mapping in the area of bioinformatics [41–46], semi-supervised classification [47], facial recognition [48], etc. Furthermore, it would be highly interesting to investigate the collaborative effects between LF combinations and specified swarm intelligence algorithms like the particle swarm optimization algorithms [49–52] and its improved versions [53–55]. These issues will be investigated in our future research.

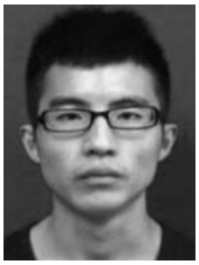
Acknowledgement

This research is supported in part by the National Key Research and Development Program of China under grant 2017YFC0804002, in part by the International Joint Project funded jointly by the Royal Society of the UK and the National Natural Science Foundation of China under grant 61611130209, in part by the National Natural Science Foundation of China under grants 61772493, 91646114, and 51609229, in part by the Pioneer Hundred Talents Program of Chinese Academy of Sciences, and in part by the Young Scientist Foundation of Chongqing under grant cstc2014kjrc-qnrc40005.

References

[1] P. Symeonidis, A. Nanopoulos, Y. Manolopoulos, Providing justifications in recommender systems, IEEE Trans. Syst. Man Cybern. A: Syst. Hum. 38 (6) (2008) 1262–1272.
[2] H.K. Kim, J.K. Kim, Y.U. Ryu, Personalized recommendation over a customer network for ubiquitous shopping, IEEE Trans. Serv. Comput. 2 (2) (2009) 140–151.
[3] Y. Koren, R. Bell, Advances in collaborative filtering, Recommender Systems Handbook, Springer US, New York, 2011, pp. 145–186.
[4] Y. Zheng, X. Xie, Learning travel recommendations from user-generated GPS traces, ACM Trans. Intell. Syst. Technol. 2 (1) (2011) 29.
[5] A. Nanopoulos, Item recommendation in collaborative tagging systems, *IEEE Trans. Syst. Man Cybern.* A: Syst. Hum. 41 (4) (2011) 760–771.
[6] W.-K. Pan, L. Chen, Group Bayesian personalized ranking with rich interactions for one-class collaborative filtering, Neurocomputing 207 (2016) 501–510.
[7] D.-S. Li, Q. Lv, L. Shang, N. Gu, Efficient privacy-preserving content recommendation for online social communities, Neurocomputing 219 (2017) 440–454.
[8] M. Balabanovic, Content-based, collaborative recommendation, Commun. ACM 40 (3) (1997) 66–72.
[9] H.-F. Liu, X.-J. Kong, X.-M. Bai, W. Wang, Context-based collaborative filtering for citation recommendation, IEEE Access 3 (2015) 1.
[10] X. Luo, M. Zhou, Z. Wang, Y. Xia, Q. Zhu, An effective QoS estimating scheme via alternating direction method-based matrix factorization, IEEE Trans. Serv. Comput. (2016), doi:10.1109/TSC.2016.2597829.
[11] X. Luo, M. Zhou, S. Li, Y. Xia, Z. You, Q. Zhu, H. Leung, Incorporation of efficient second-order solvers into latent factor models for accurate prediction of missing QoS data, IEEE Trans. Cybern. (2017), doi:10.1109/TCYB.2017.2685521.
[12] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, IEEE Trans. Knowl. Data Eng. 17 (6) (2005) 734–749.
[13] B. Prasad, HYREC: a hybrid recommendation system for e-commerce, in: Proceedings of the 6th International Conference on Case-Based Reasoning, USA, Chicago, Aug. 2005, pp. 23–26.
[14] J.-M. Yang, K.-F. Li, Recommendation based on rational inferences in collaborative filtering, Knowl.-Based Syst. 22 (1) (2009) 105–114.
[15] A. Gogna, A. Majumdar, A comprehensive recommender system model: improving accuracy for both warm and cold start users, IEEE Access 3 (2015) 2803–2813.
[16] I. Im, B.H. Kim, Personalizing the settings for CF-based recommender systems, in: Proceedings of the 4th ACM Conference on Recommender Systems, Spain, Barcelona, Sep. 2010, pp. 245–248.
[17] K. Junzo, A. Tomofumi, S. Shinji, M. Hideo, A community-based recommendation system to reveal unexpected interests, in: Proceedings of the 11th International Conference on Multimedia Modelling, Washington, USA, Jan. 2005, pp. 433–438.
[18] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web, Hong Kong, May. 2001, pp. 285–295.
[19] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, IEEE Comput. 42 (8) (2009) 30–37.
[20] T. Hofmann, Latent semantic models for collaborative filtering, ACM Trans. Inform. Syst. 22 (1) (2004) 89–115.
[21] N. Srebro, J.D.M. Rennie, T. Jaakkola, Maximum-margin matrix factorization, Adv. Neural Inf. Process. Syst. 37 (2) (2004) 1329–1336.
[22] G. Takács, I. Pilászy, B. Németh, D. Tikk, Scalable collaborative filtering approaches for large recommender systems, J. Mach. Learn. Res. 10 (10) (2009) 623–656.
[23] Y. Koren, Factorization meets the neighborhood: A multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Neveda, 2008, pp. 426–434.
[24] X. Luo, M. Zhou, M. Shang, S. Li, Y. Xia, A novel approach to extracting non-negative latent factors from non-negative big sparse matrices, IEEE Access 4 (2016) 1.
[25] K. Fukunaga, L. Hostetler, k-nearest-neighbor Bayes-risk estimation, IEEE Trans. Inf. Theory 21 (3) (1975) 285–293.
[26] X. Luo, M. Zhou, S. Li, Z. You, A nonnegative latent factor model for large-scale sparse matrices in recommender systems via alternating direction method, IEEE Trans. Neural Netw. Learn. Syst. 27 (3) (2015) 579–592.
[27] X. Luo, M.C. Zhou, S. Li, Y. Xia, Z. You, Q. Zhu, H. Leung, An efficient second-order approach to factorize sparse matrices in recommender systems, IEEE Trans. Ind. Inform. 11 (4) (2015) 1.
[28] M. Rossetti, F. Stella, M. Zanker, Towards explaining latent factors with topic models in collaborative recommender systems, in: Proceedings of the 24th Int. Workshop Conference on Database and Expert Systems Applications, Sep. 2013, pp. 26–30.
[29] D.P. Bertsekas, Nonlinear programming, J. Oper. Res. Soc. 48 (3) (1997) 334.
[30] S. Boyd, L. Vandenberghe, Convex Optimization, IEEE Trans. Autom. Control 51 (11) (2006) 1859.
[31] T. Gábor, P. István, N. Bottyán, T. Domonkos, Investigation of various matrix factorization methods for large recommender systems, in: Proceedings of the 8th IEEE International Conference on Data Mining Workshops, Las Vegas, Nevada, Dec. 2008, pp. 553–562.
[32] A. Mnih, R. Salakhutdinov, Probabilistic matrix factorization, in: Proceedings of the 22nd International Conference on Machine Learning, 2012, pp. 880–887.

[33] X. Luo, M. Zhou, Y. Xia, Q. Zhu, An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems, IEEE Trans. Ind. Inform. 10 (2) (2014) 1273–1284.
[34] X. Luo, M. Zhou, Y. Xia, Q. Zhu, A.C. Ammari, A. Alabdulwahab, Generating highly accurate predictions for missing QoS data via aggregating nonnegative latent factor models, IEEE Trans. Neural Netw. Learn. Syst. 27 (3) (2015) 524–537.
[35] X. Luo, Y. Xia, Q. Zhu, Incremental collaborative filtering recommender based on regularized matrix factorization, Knowl.-Based Syst. 27 (3) (2011) 271–280.
[36] X. Luo, M. Zhou, H. Leung, Y. Xia, An incremental-and-static-combined scheme for matrix-factorization-based collaborative filtering, IEEE Trans. Autom. Sci. Eng. 13 (1) (2014) 1–11.
[37] J.L. Herlocker, Evaluating collaborative filtering recommender systems, ACM Trans. Inf. Syst. 22 (1) (2004) 5–53.
[38] M. Friedman, A comparison of alternative tests of significance for the problem of m rankings, Ann. Math. Stat. 11 (1) (1940) 86–92.
[39] T. Wang, W. Zhang, C. Ye, J. Wei, FD4C: automatic fault diagnosis framework for web applications in cloud computing, IEEE Trans. Syst. Man Cybern.: Syst. 46 (1) (2016) 61–75.
[40] Y. Xun, J. Zhang, X. Qin, FiDoop: parallel mining of frequent itemsets using MapReduce, IEEE Trans. Syst. Man Cybern.: Syst. 46 (3) (2016) 313–325.
[41] Z.-H. You, Y.-K. Lei, D.-S. Huang, X.-B. Zhou, Using manifold embedding for assessing and predicting protein interactions from high-throughput experimental data, Bioinformatics 26 (21) (2010) 2744–2751.
[42] Z.-H. You, Z. Yin, K. Han, D.-S. Huang, X.-B. Zhou, A semi-supervised learning approach to predict synthetic genetic interactions by combining functional and topological properties of functional gene network, BMC Bioinform. 11 (1) (2010) 343–351.
[43] Z.-H. You, J.-Z. Yu, L. Zhu, S. Li, Z.-K. Wen, A MapReduce based parallel SVM for large-scale predicting protein-protein interactions, Neurocomputing 145 (5) (2014) 37–43.
[44] Z.-H. You, L. Zhu, C.-H. Zheng, H.-J. Yu, S.-P Deng, Z. Ji, Prediction of protein-protein interactions from amino acid sequences using a novel multi-scale continuous and discontinuous feature set, BMC Bioinform. 15 (15) (2014) 9.
[45] Z.-H. You, M.-C. Zhou, X. Luo, S. Li, Highly efficient framework for predicting interactions between proteins, IEEE Trans. Cybern. 47 (3) (2016) 721–733.
[46] Z.-H. You, Y.-K. Lei, L. Zhu, J.-F. Xia, B. Wang, Prediction of protein-protein interactions from amino acid sequences with ensemble extreme learning machines and principal component analysis, BMC Bioinform. 14 (8) (2013) 10.
[47] D. Wu, X. Luo, G.-Y. Wang, M.-S. Shang, Y. Yuan, H.-Y. Yan, A highly-accurate framework for self-labeled semi-supervised classification in industrial applications, IEEE Trans. Ind. Inform. (2017), doi:10.1109/TII.2017.2737827.
[48] F. Wang, B. Shen, S. Sun, Z. Wang, Improved GA and Pareto optimization-based facial expression recognition, Assem. Autom. 36 (2) (2016) 192–199.
[49] N. Zeng, H. Zhang, W. Liu, J. Liang, F.E. Alsaadi, A switching delayed PSO optimized extreme learning machine for short-term load forecasting, Neurocomputing 240 (2) (2011) 175–182.
[50] N. Zeng, Z. Wang, Y. Li, M. Du, X. Liu, A hybrid EKF and switching PSO algorithm for joint state and parameter estimation of lateral flow immunoassay models, IEEE/ACM Trans. Comput. Biol. Bioinform. 9 (2) (2017) 321–329.
[51] N. Zeng, Z. Wang, Y. Li, M. Du, X. Liu, Inference of nonlinear state-space models for sandwich-type lateral flow immunoassay using extended Kalman filtering, IEEE Trans. Biomed. Eng. 58 (7) (2011) 1959–1966.
[52] N. Zeng, Z. Wang, Y. Li, M. Du, X. Liu, Identification of nonlinear lateral flow immunoassay state-space models via particle filter approach, IEEE Trans. Nanotechnol. 11 (2) (2012) 321–327.
[53] N. Zeng, Z. Wang, Y. Li, M. Du, J. Cao, X. Liu, Time series modeling of nano-gold immunochromatographic assay via expectation maximization algorithm, IEEE Trans. Biomed. Eng. 60 (12) (2013) 3418–3424.
[54] N. Zeng, Y.S. Hung, Y. Li, M. Du, A novel switching local evolutionary PSO for quantitative analysis of lateral flow immunoassay, Expert Syst. Appl. 41 (4) (2014) 1708–1715.
[55] N. Zeng, Z. Wang, B. Zineddin, Y. Li, M. Du, L. Xiao, X. Liu, T. Young, Image-based quantitative analysis of gold immunochromatographic strip via cellular neural network approach, IEEE Trans. Med. Imaging 33 (5) (2014) 1129–1136.



Ye Yuan received the B.S. degree in electronic information engineering and the M.S. degree in signal processing from the University of Electronic Science and Technology, Chengdu, China, in 2010 and 2013, respectively. He is currently working toward the Ph.D. degree in computer science from Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing, China. His research interests include data mining, recommender system, intelligent computing, and their applications.

Xin Luo received the B.S. degree in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 2005, and the Ph.D. degree in computer science from Beihang University, Beijing, China, in 2011. He joined the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing, China in 2016, and is currently a professor of computer science and engineering. His research interests include big data analysis and intelligent control. He has published 70+ papers (including 20 IEEE TRANSACTIONS papers) in his related areas.

Ming-Sheng Shang received his Ph.D. degree in Computer Science from University of Electronic Science and Technology of China in Chengdu (UESTC), China in 2007. He joined the School of Computer Science and Engineering, UESTC, Chengdu, China, in 2002, and was a Professor with UESTC. He is currently a Professor at the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing, China, and served as deputy director of Institute of Electronic Information Technology, director of Big Data Mining Center. His research interests are in complex network analysis and big data applications.