An L_1 -and- L_2 -Norm-Oriented Latent Factor Model for Recommender Systems

Di Wu¹⁰, Member, IEEE, Mingsheng Shang¹⁰, Xin Luo¹⁰, Senior Member, IEEE,

and Zidong Wang^(D), *Fellow*, *IEEE*

Abstract—A recommender system (RS) is highly efficient in filtering people's desired information from high-dimensional and sparse (HiDS) data. To date, a latent factor (LF)-based approach becomes highly popular when implementing a RS. However, current LF models mostly adopt single distance-oriented Loss like an L_2 norm-oriented one, which ignores target data's characteristics described by other metrics like an L_1 normoriented one. To investigate this issue, this article proposes an L_1 -and- L_2 -norm-oriented LF (L³F) model. It adopts twofold ideas: 1) aggregating L_1 norm's robustness and L_2 norm's stability to form its Loss and 2) adaptively adjusting weights of L_1 and L_2 norms in its Loss. By doing so, it achieves fine aggregation effects with L_1 norm-oriented Loss's robustness and L₂ norm-oriented Loss's stability to precisely describe HiDS data with outliers. Experimental results on nine HiDS datasets generated by real systems show that an L³F model significantly

Manuscript received November 6, 2019; revised May 19, 2020, October 26, 2020, and February 8, 2021; accepted March 31, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61702475, Grant 61772493, Grant 62002337, and Grant 61902370; in part by the CAAI-Huawei MindSpore Open Fund under Grant CAAIXSJLJJ-2020-004B; in part by the Natural Science Foundation of Chongqing (China) under Grant cstc2019jeyj-msxmX0578 and Grant cstc2019jeyjiqX0013; in part by the Chinese Academy of Sciences "Light of West China" Program; in part by the Technology Innovation and Application Development Project of Chongqing, China, under Grantcstc2018jszx-cyzdX0041 and Grant cstc2019jsx-fxydX0027; and in part by the Pioneer Hundred Talents Program of Chinese Academy of Sciences. (*Di Wu and Xin Luo are co-first authors.*) (*Corresponding author: Xin Luo*).

Di Wu is with the Chongqing Engineering Research Center of Big Data Application for Smart Cities, Chinese Academy of Sciences, Chongqing 400714, China, also with the Chongqing Key Laboratory of Big Data and Intelligent Computing, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China, and also with the Chongqing School, University of Chinese Academy of Sciences, Chongqing 400714, China (e-mail: wudi@cigit.ac.cn).

Mingsheng Shang is with the Chongqing Engineering Research Center of Big Data Application for Smart Cities, Chinese Academy of Sciences, Chongqing 400714, China, and also with the Chongqing Key Laboratory of Big Data and Intelligent Computing, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China (e-mail: msshang@cigit.ac.cn).

Xin Luo is with the Chongqing Engineering Research Center of Big Data Application for Smart Cities, Chinese Academy of Sciences, Chongqing 400714, China, also with the Chongqing Key Laboratory of Big Data and Intelligent Computing, Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 400714, China, also with the Department of Big Data Analyses Techniques, Cloudwalk, Chongqing 401331, China, and also with the Chongqing School, University of Chinese Academy of Sciences, Chongqing 400714, China (e-mail: luoxin21@cigit.ac.cn).

Zidong Wang is with the Department of Computer Science, Brunel University London, Uxbridge UB8 3PH, U.K. (e-mail: Zidong.Wang@brunel.ac.uk). This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TNNLS.2021.3071392.

Digital Object Identifier 10.1109/TNNLS.2021.3071392

outperforms state-of-the-art models in prediction accuracy for missing data of an HiDS dataset. Its computational efficiency is also comparable with the most efficient LF models. Hence, it has good potential for addressing HiDS data from real applications.

Index Terms—High-dimensional and sparse (HiDS) matrix, latent factor (LF) analysis, L_1 norm, L_2 norm, recommender system (RS).

I. INTRODUCTION

I N THIS era of information explosion, people are inundated by big data [1]. For instance, Google's data come to PBs and Flickr generates TBs every day [2]. How to implement an intelligent system to filter desired information out of such big data is highly challenging [1], [3], [4]. Recommender system (RS) is highly useful in addressing this issue [5], [6]. So far, various approaches are proposed to implement an RS, where collaborative filtering (CF) is highly popular [7]–[13].

Great efforts have been made to achieve various CF-based RSs, where a latent factor (LF) model [14]–[16] is widely adopted owing to its high efficiency and scalability in industrial applications [17], [18]. Commonly, an LF model is developed based on a user-item rating matrix [3], [19], where each row denotes a specific user, each column denotes a specific item (e.g., movie, electronic product, and music), and each entry denotes a user's preference on an item. Note that a user cannot touch all items since the item count can be huge in an industrial RS like Amazon [20]. Thus, a user-item matrix is commonly high-dimensional and sparse (HiDS).

Given an HiDS matrix, an LF model maps both users and items into the same low-dimensional LF space to train the desired LFs based on the observed entries only, and then estimate its missing entries relying heavily on these trained LFs [9]. An LF model's objective function commonly has the form of *Loss* + *Penalty* [18], [21] that should be carefully designed. On an HiDS matrix, the *Loss* is the sum error between its observed entries and corresponding estimates generated by an LF model, and the *Penalty* commonly consists of regularization terms to prevent an LF model from overfitting.

Considering the *Loss* of an LF model, it usually depends on L_1 or L_2 norm defined on the known data of an HiDS matrix. Fig. 1 illustrates the differences between L_1 norm-oriented and L_2 norm-oriented *Losses*.

1) The former is less sensitive to outliers than the latter, thereby enhancing the robustness of a resultant model [22]–[25] as shown in Fig. 1(a).

2162-237X © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.



Fig. 1. Differences between L_1 and L_2 norms. (a) Fitness on ten instances with two outliers. (b) Loss functions with L_1 norm and L_2 norm, respectively, when the difference between predictions and ground truth is small (less than 1).

2) The latter is smoother than the former when the predictions and ground truth data are close, thereby enhancing the stability of a resultant model [26] as shown in Fig. 1(b).

An HiDS matrix generated by real RSs are commonly filled with outliers by malicious users (e.g., a user always assigns extremely high or low raring to an item) [27], [28], which can greatly impair the performance of an LF model relying on L_2 norm-oriented *Loss* only. On the other hand, an LF model relying on L_1 norm-oriented *Loss* solely suffers from unstable prediction performance, which can greatly harm its overall prediction accuracy for a user's unknown rating on an item. From this point of view, an LF model with a *Loss* relying on L_1 or L_2 norm only fails in well describing an HiDS matrix [25], [29].

To the authors' best knowledge, an existing LF model's *Loss* commonly depends solely on L_1 norm, L_2 norm, or other distance metrics like Kullback-Leibler Divergence [26], [27], [8], [30]–[34]. Related studies [35]–[40] also propose to combine both L_1 and L_2 norms in the *Penalty* rather than *Loss* of an LF model. How will an LF model perform with its *Loss* relying on multiple norms which are efficiently aggregated? What is the theoretical evidence behind its performance? Motivated by these critical issues, this study proposes an L_1 -and- L_2 -normoriented LF (L³F) model. It is significantly different from existing LF models owing to its L_1 -and- L_2 -normoriented *Loss* with an adaptive weighting strategy for well aggregating the effects of both L_1 and L_2 norms. It achieves both robustness and stability to well describe an HiDS matrix from an RS. Main contributions of this study include:

- An L³F model is proposed. It efficiently aggregates L₁ norm-oriented *Loss*'s robustness and L₂ norm-oriented *Loss*'s stability to well describe an HiDS rating matrix. Hence, it is robust to outliers in an HiDS rating matrix, as well as achieves high prediction accuracy for missing data of an HiDS rating matrix.
- 2) Theoretical analyses and proofs showing an $L^{3}F$ model's ability to aggregate the effects of L_{1} and L_{2} norm-oriented *Losses* are presented.
- 3) Algorithm design and analysis for an $L^{3}F$ model.

Empirical studies on nine HiDS matrices generated by real RSs are carefully conducted to evaluate L^3F 's performance. Results demonstrate that compared with state-of-the-art LF models, an L^3F model achieves significant accuracy gain when predicting missing data of an HiDS rating matrix.

Its computational efficiency is also highly competitive when compared with the most efficient LF models.

Note that an RS is a learning system highly efficient in filtering people's desired information out from big data [3], [5], [6]. The L³F proposed by this study can promote the development and applications of related RSs [14]–[16]. Moreover, this study shows that L³F can well represent HiDS data with outliers by aggregating L_1 norm-oriented *Loss*'s robustness and L_2 norm-oriented *Loss*'s stability [22], [23], [26]. Its principle can be adopted to address similar issues raised by neural network-based learning systems [2], [50], [51], [58].

Section II gives the preliminaries. Section III proposes an L³F model. Section IV presents empirical studies. Finally, Section V concludes this article.

II. PRELIMINARIES

A. Symbols and Notations

Please refer to Table S.I in the Supplementary File.

B. Related Work

An LF model is widely adopted to implement an RS [17], [18]. So far, various sophisticated LF models have been proposed, including a bias-based one [18], a nonparametric one [31], a nonnegativity-constrained one [19], a probabilistic one [30], a dual-regularization-based one [32], a posterior-neighborhood-regularized one [41], a randomized one [42], a graph regularized one [43], a neighborhood-andlocation integrated one [8], a confidence-driven one [33], and a data characteristic-aware one [34]. Although they are different from each other in objective functions or learning algorithms, they all adopt an L_2 norm-oriented Loss that is highly sensitive to outliers [25], [27], [28]. To make an LF model less sensitive to outlier data, Zhu et al. [24] propose to adopt an L_1 -norm-oriented Loss. However, an LF model with an L_1 norm-oriented Loss has possibly multiple solution spaces because L_1 norm is less smooth than L_2 norm.

On the other hand, matrix completion [35]–[38] or feature representation [39], [40] models adopt both L_1 and L_2 norms to construct their *Penalty*, thereby achieving model sparsity [44] or generality [37]. Nonetheless, as mentioned before, *Penalty* and *Loss* are two different and critical components of an LF model's learning objective. *Penalty* affects a resultant model's own characteristics like model sparsity, while *Loss* mostly decides how an achieved model describes target data. But different from existing models, an L³F model investigates the effects by multiple norms-oriented *Loss*. It implements an L_1 -and- L_2 -norm-oriented *Loss* where the effects of L_1 and L_2 norms are aggregated efficiently via an adaptive weighting strategy, thereby achieving both robustness to outliers and model stability simultaneously.

Recently, deep neural networks (DNN) [45]-based approaches to an RS attract researchers' attention [46]–[49]. Zhang *et al.* [50] conduct a detailed review of DNN-based RSs. Sophisticated DNN-based recommender models include an autoencoder-based one [51], a hybrid autoencoder-based one [52], a multitask learning-oriented one [53], a neural factorization-based one [54], an attentional factorization-based

one [55], a deep cooperative neural network model [56], and a convolutional matrix factorization model [57]. However, when addressing HiDS data, a DNN-based model's performance is achieved with high computation burden [58]–[60], while an $L^{3}F$ model does not suffer from such limitations. Note that in Section IV-E, some DNN-based models [54]–[57] mentioned above are not compared since they are defined on different data sources. More specifically, models proposed in [54] and [55] focus on implicit feedbacks like log information rather than explicit feedbacks like ratings concerned in this study. Models proposed in [56], [57] rely on additional review information.

C. Problem Definition

In our context, an HiDS matrix is defined as in [9], [16], and [18]

Definition 1: Given a user set U and an item set I, $Y^{|U| \times |I|}$ is a matrix where each element $y_{u,i}$ describes user $u \in U$'s preference on item $i \in I$. Let R_K and R_U denote its known and unknown entry sets, respectively. Y is HiDS if $|R_K| \ll |R_U|$.

Given Y, an LF model is built on R_K as defined in [17] and [18].

Definition 2: Given Y and f, an LF model aims to achieve LF matrices $P^{|U| \times f}$ and $Q^{|I| \times f}$ for Y's rank-f approximation $\hat{Y} = PQ^{T}$ based on R_{K} only with $f \ll \min\{|U|, |I|\}$.

Thus, an objective function defined on R_K is highly desired to achieve P and Q. It commonly has the form of *Loss* + *Penalty* with P and Q [18], where *Loss* quantifies the distance between Y and \hat{Y} and *Penalty* generalizes the achieved model. So the learning objective of an LF model is given as

$$\underset{P,Q}{\operatorname{arg\,min}} \varepsilon(P, Q) = L(P, Q) + \lambda F(P, Q) \tag{1}$$

where L(P, Q) is the Loss, F(P, Q) is the Penalty, and λ is the constant adjusting the Penalty effects, respectively. Most existing LF models adopt a unique norm-oriented Loss like an L_2 norm-oriented one with linear bias [18]

$$L_{2}(P, Q) = \|\Omega \odot (Y - W - \hat{Y})\|_{L_{2}}^{2}$$

= $\|\Omega \odot (Y - W - PQ^{T})\|_{L_{2}}^{2}$ (2)

where \odot denotes the Hadamard product performing the element-wise multiplication between two matrices, *W* is a linear bias matrix consisting of linear bias related with $\forall y_{u,i} \in R_K$, and Ω is a $|U| \times |I|$ binary indexing matrix given as

$$\Omega_{u,i} = \begin{cases} 1, & \text{if } y_{u,i} \in R_K \\ 0, & \text{otherwise.} \end{cases}$$
(3)

Note that L_2 norms in (2) can be replaced with L_1 norms as

$$L_1(P, Q) = \left\| \Omega \odot \left(Y - W - \hat{Y} \right) \right\|_{L_1}$$

= $\left\| \Omega \odot \left(Y - W - P Q^T \right) \right\|_{L_1}.$ (4)

Considering F(P, Q), it often depends on the L_2 norm of P and Q to enhance an LF model's generality:

$$F(P,Q) = \|P\|_{L_2}^2 + \|Q\|_{L_2}^2.$$
 (5)

III. L_1 -AND- L_2 -NORM-ORIENTED LF MODEL

A. Objective Formulation

As shown above, either L_1 or L_2 norm-oriented *Loss* has its own advantage in describing HiDS data [25], [29]. To aggregate their effects, an L_1 -and- L_2 -norm-oriented *Loss* is built as

$$\arg\min_{\substack{P,Q\\P,Q}} \varepsilon(P, Q) = \alpha_1 \| \Omega \odot (Y - W - PQ^{\mathrm{T}}) \|_{L_1} + \alpha_2 \| \Omega \odot (Y - W - PQ^{\mathrm{T}}) \|_{L_2}^2 + \lambda (\|P\|_{L_2}^2 + \|Q\|_{L_2}^2)$$
(6)

where α_1 and α_2 are aggregation weights controlling the effects of L_1 and L_2 norms, respectively. Note that we make $\alpha_1 + \alpha_2 = 1$ and $\alpha_1, \alpha_2 \ge 0$ to maintain the numerical magnitude of the *Loss*. Moreover, due to *Y*'s extreme sparsity and R_K 's imbalance (e.g., some users are related with many ratings and the others are related with few), it is vital to expand (6) into a density-oriented form [17], [19]

where $p_{u,d}$ and $q_{i,d}$ denote specific entries in *P* and *Q*, $w_{u,i}$ is given by $w_{u,i} = \mu + b_u + b_i$ as μ denotes the global-average of R_K , b_u denotes the observed deviations on user *u*, and b_i denotes the observed deviations on item *I*, respectively. Note that in (7), the regularization effect on each LF is specified with its relevant known rating count [17], [19], thereby implementing a finely grained control of the regularization effects.

With (7), an L³F model reasonably aggregates the merits of L_1 norm-based and L_2 norm-oriented *Losses*, i.e., model robustness and stability. Note that when $w_{u,i} = 0$, (7) involves no linear bias, which can be considered as a special case of an L³F model. Hereafter, we mark models with and without linear bias as L³F_b and L³F_b.

B. Model Optimization

The optimization of (7) with *P* and *Q* can be achieved by various learning algorithms. As discussed in [17] and [19], a stochastic gradient descent (SGD) algorithm is efficient to do so. It considers the instant loss on a single rating $y_{u,i}$ in (7), as shown at the bottom of the next page, as

$$\varepsilon_{u,i} = \alpha_1 \left| y_{u,i} - w_{u,i} - \sum_{d=1}^f p_{u,d} q_{i,d} \right| + \alpha_2 \left(y_{u,i} - w_{u,i} - \sum_{d=1}^f p_{u,d} q_{i,d} \right)^2 + \lambda \left(\sum_{d=1}^f (p_{u,d})^2 + \sum_{d=1}^f (q_{i,d})^2 \right).$$
(8)

Then, in the *n*th iteration, it moves each single LF along the opposite direction of the stochastic gradient of (8) with it as

$$\forall d \in \{1, 2, \dots, f\}: \begin{cases} p_{u,d}^n = p_{u,d}^{n-1} - \eta \frac{\partial \varepsilon_{u,i}^{n-1}}{\partial p_{u,d}^{n-1}} \\ q_{i,d}^n = q_{i,d}^{n-1} - \eta \frac{\partial \varepsilon_{u,i}^{n-1}}{\partial q_{i,d}^{n-1}} \end{cases}$$
(9)

where $\varepsilon_{u,i}^{n-1}$, $p_{u,d}^{n-1}$, and $q_{i,d}^{n-1}$ denote the states of ε , $p_{u,d}$, and $p_{i,d}$ during the (n-1)th iteration, and η denotes the learning rate, respectively. Let $\Delta_{u,i} = y_{u,i} - w_{u,i} - \sum_{d=1}^{f} p_{u,d}q_{i,d}$, then is formulated as the expression of (8) actually depends on the sign of $\Delta_{u,i}$

$$\begin{cases} \Delta_{u,i}^{n-1} \ge 0 : \begin{cases} \varepsilon_{u,i}^{n-1} = \alpha_1^{n-1} \Delta_{u,i}^{n-1} + \alpha_2^{n-1} \left(\Delta_{u,i}^{n-1} \right)^2 \\ + \lambda \left(\sum_{d=1}^f \left(p_{u,d}^{n-1} \right)^2 + \sum_{d=1}^f \left(q_{i,d}^{n-1} \right)^2 \right) \\ \varepsilon_{u,i}^{n-1} = -\alpha_1^{n-1} \Delta_{u,i}^{n-1} + \alpha_2^{n-1} \left(\Delta_{u,i}^{n-1} \right)^2 \\ + \lambda \left(\sum_{d=1}^f \left(p_{u,d}^{n-1} \right)^2 + \sum_{d=1}^f \left(q_{i,d}^{n-1} \right)^2 \right) \end{cases}$$
(10)

where α_1^{n-1} , α_2^{n-1} , and $\Delta_{u,i}^{n-1}$ are the states of α_1 , α_2 , and $\Delta_{u,i}$ during the (n-1)th iteration, respectively.

By combining (9) and (10), we have the following scheme:

On
$$\mathbf{y}_{u,i} \quad \forall d \in \{1, 2, ..., f\}$$
:

$$\begin{cases}
\Delta_{u,i}^{n-1} > 0: \begin{cases}
p_{u,d}^{n} = p_{u,d}^{n-1} + \alpha_{1}^{n-1} \eta q_{i,d}^{n-1} + \alpha_{2}^{n-1} \eta q_{i,d}^{n-1} \Delta_{u,i}^{n-1} \\
-\eta \lambda p_{u,d}^{n-1} \\
q_{i,d}^{n} = q_{i,d}^{n-1} + \alpha_{1}^{n-1} \eta p_{u,d}^{n-1} + \alpha_{2}^{n-1} \eta p_{u,d}^{n-1} \Delta_{u,i}^{n-1} \\
-\eta \lambda q_{i,d}^{n-1} \\
-\eta \lambda q_{i,d}^{n-1} \\
q_{i,d}^{n} = p_{u,d}^{n-1} - \alpha_{1}^{n-1} \eta q_{i,d}^{n-1} + \alpha_{2}^{n-1} \eta q_{i,d}^{n-1} \Delta_{u,i}^{n-1} \\
-\eta \lambda p_{u,d}^{n-1} \\
q_{i,d}^{n} = q_{i,d}^{n-1} - \alpha_{1}^{n-1} \eta p_{u,d}^{n-1} + \alpha_{2}^{n-1} \eta p_{u,d}^{n-1} \Delta_{u,i}^{n-1} \\
-\eta \lambda p_{u,d}^{n-1} \\
q_{i,d}^{n} = q_{i,d}^{n-1} - \alpha_{1}^{n-1} \eta p_{u,d}^{n-1} + \alpha_{2}^{n-1} \eta p_{u,d}^{n-1} \Delta_{u,i}^{n-1} \\
-\eta \lambda q_{i,d}^{n-1}.
\end{cases}$$
(11)

C. Self-Adaptive Aggregation and Proof

To finely aggregate the effects of L_1 and L_2 norm-oriented Losses, we make α_1 and α_2 adaptive according to the training error. Let L_1^n and L_2^n denote the partial loss depending on L_1 and L_2 norms in (7) at the *n*th iteration, respectively. The main idea is to increase α_1 and decrease α_2 if $L_1^n < L_2^n$, and decrease α_1 and increase α_2 otherwise. For theoretically validating the effectiveness of this strategy, we firstly present the following definitions:

Definition 3: Let L_1^n and L_2^n be the states of partial Losses separately depending on L_1 and L_2 norms in (7) at the *n*th training iteration, then L_1^n and L_2^n are given as

$$\forall u \in \{1, 2, \dots, |U|\} \quad \forall i \in \{1, 2, \dots, |I|\}:$$
$$L_1^n = \sum_{(u,i)\in R_K} |\Delta_{u,i}^n|, \quad L_2^n = \sum_{(u,i)\in R_K} (\Delta_{u,i}^n)^2. \quad (12)$$

Definition 4: Let L_{12}^n be the state of L³F's Loss based on aggregating L_1^n and L_2^n in the *n*th training iteration, then L_{12}^n

IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS

$$\forall u \in \{1, 2, \dots, |U|\} \quad \forall i \in \{1, 2, \dots, |I|\} : L_{12}^n = \alpha_1^n \sum_{(u,i) \in R_K} |\Delta_{u,i}^n| + \alpha_2^n \sum_{(u,i) \in R_K} (\Delta_{u,i}^n)^2 = \alpha_1^n L_1^n + \alpha_2^n L_2^n.$$
(13)

Definition 5: Given L_1^n , L_2^n , and L_{12}^n , let C_{L1}^n , C_{L2}^n , and C_{L1}^n be the cumulative Loss corresponding to L_1^n , L_2^n , and L_{12}^n till the *n*th iteration, respectively. Then C_{L1}^n , C_{L2}^n , and C_{L12}^n are given as

$$C_{L_1}^n = \sum_{j=1}^n L_1^j, \quad C_{L_2}^n = \sum_{j=1}^n L_2^j, \quad C_{L_{12}}^n = \sum_{j=1}^n L_{12}^j.$$
 (14)

Based on *Definitions* 3–5, we present *Theorem 1:*

Theorem 1: Considering an L³F model, assuming that its L_1^n and L_2^n lie in the scale of [0, 1]. If α_1^n and α_2^n fulfill the following condition:

$$\alpha_1^n = \frac{e^{-\gamma C_{L_1}^{n-1}}}{e^{-\gamma C_{L_1}^{n-1}} + e^{-\gamma C_{L_2}^{n-1}}}, \quad \alpha_2^n = \frac{e^{-\gamma C_{L_2}^{n-1}}}{e^{-\gamma C_{L_1}^{n-1}} + e^{-\gamma C_{L_2}^{n-1}}} \quad (15)$$

then the following equality holds:

$$C_{L_{12}}^{N} \le \min\{C_{L_{1}}^{N}, C_{L_{2}}^{N}\} + \frac{\ln 2}{\gamma} + \frac{\gamma N}{8}.$$
 (16)

Note that in **Theorem 1**, α_1^n and α_2^n denote the states of α_1 and α_2 in the *n*th training iteration, and γ denotes a hyperparameter controlling their learning rates. More specifically, with $\gamma = (1/\ln N)^{1/2}$, the upper bound becomes min $\{C_{L1}^N, C_{L2}^N\} + \ln 2(\ln N)^{1/2} + N/(8(\ln N)^{1/2})$. Note that the term of $\ln 2(\ln N)^{1/2} + N/(8(\ln N)^{1/2})$ is linearly bounded by the number of iterations. Hence, C_{L1}^N is comparable to the minimum of C_{L1}^N and C_{L2}^N after N training iterations. Based on *Theorem* 1, we present the following proposition:

Proposition 1: Given that $\gamma = (1/\ln N)^{1/2}$, if $C_{L1}^N > C_{L2}^N$, the following inequality holds:

$$C_{L_{12}}^{N} \le C_{L_{2}}^{N} + \text{const} < C_{L_{1}}^{N} + \text{const}$$
 (17)

otherwise if $C_{L1}^N < C_{L2}^N$, the following inequality holds:

$$C_{L_{12}}^N \le C_{L_1}^N + \text{const} < C_{L_2}^N + \text{const}$$
 (18)

where $\lim_{N\to\infty} \text{const} = 19.45$.

Remark 1: Proposition 1 states that C_{L12}^N is bounded by C_{L1}^N + const and C_{L2}^N + const with the condition of $\gamma = (1/\ln N)^{1/2}$. Hence, an L³F model's cumulative prediction





error is always comparable to (or not larger than) that of an LF model solely built on an L_1 or L_2 norm-oriented *Loss* during the training process.

Note that to enable an $L^{3}F$ model's practicability, the balancing coefficient γ is set self-adaptive as follows:

$$\gamma^{n} = \frac{1}{C_{L_{1}}^{n} + C_{L_{2}}^{n}}$$
(19)

where γ^n indicates the state of γ in the *n*th training iteration.

Section III-C1 gives the sketch proofs of *Theorem* 1 and *Proposition* 1 for conciseness. Note that their complete proofs are provided in the Supplementary File of this article.

1) Proof of Theorem 1: Firstly, recall the Hoeffding Inequality [61].

Lemma 1: Let *X* be a random variable fulfilling $a \le X \le b$, $\forall s \in \mathbb{R}$ the following inequality holds:

$$\ln \mathbb{E}\left[e^{sX}\right] \le s\mathbb{E}X + \frac{s^2(b-a)^2}{8}.$$
(20)

Note that the detailed proof of *Lemma* 1 is given in [62]. \Box

Then note that C_{L1}^N and C_{L2}^N are bounded by

$$A_n = e^{-\gamma C_{L_1}^n} + e^{-\gamma C_{L_2}^n}$$
(21)

and $A_0 = 1 + 1 = 2$ since there is no *Loss* when n = 0. Based on (21), we have the following inference:

$$\ln \frac{A_N}{A_0} \ge -\gamma \min \{ C_{L_1}^N, C_{L_2}^N \} - \ln 2.$$
 (22)

Besides, $\forall n \in \{1, \dots, N\}$, based on (15) and (21), we have

$$\ln \frac{A_n}{A_{n-1}} = \ln \left(\alpha_1^n e^{-\gamma L_1^n} + \alpha_2^n e^{-\gamma L_2^n} \right).$$
(23)

Let $s = -\gamma$ and $X \in \{L_1^n, L_2^n\}$ with L_1^n and L_2^n in the scale of [0, 1]. Note that α_1^n and α_2^n are interpreted as the probabilities for L_1^n and L_2^n , respectively. Thus, we have

$$sEX = -\gamma \left(\alpha_1^n L_1^n + \alpha_2^n L_2^n \right).$$
⁽²⁴⁾

Then based on Lemma 1, (13), (23), and (24), we have

$$\ln \frac{A_n}{A_{n-1}} \le -\gamma \left(\alpha_1^n L_1^n + \alpha_2^n L_2^n \right) + \frac{\gamma^2}{8} = -\gamma L_{12}^n + \frac{\gamma^2}{8}.$$
 (25)

Considering the accumulation of (25) as *n* increases from 1 to *N*, we have the following inferences:

$$\ln \frac{A_N}{A_0} \le -\gamma C_{L_{12}}^N + \frac{\gamma^2}{8}N.$$
 (26)

By combining (22) and (26), the following inequality is achieved:

$$C_{L_{12}}^N \le \min\{C_{L_1}^N, C_{L_2}^N\} + \frac{\ln 2}{\gamma} + \frac{\gamma N}{8}.$$
 (27)

Based on (27), *Theorem* 1 holds. □

2) Proof of Proposition 1: Firstly, recall the Bernstein Inequality [62].

Lemma 2: Let X be a random variable in the scale of [0, 1], then $\forall s \in \mathbb{R}$ the following inequality holds:

$$\ln E[e^{sX}] \le (e^s - 1)EX.$$
(28)

Note that the detailed proof of *Lemma* 2 is given in [62]. \Box When $C_{L1}^N > C_{L2}^N$, we reduce (22) into

$$\ln \frac{A_N}{A_0} \ge -\gamma \, C_{L_2}^N - \ln 2. \tag{29}$$

By combining *Lemma* 2, (13), (23), and (24), the following inequality is achieved:

$$\ln \frac{A_n}{A_{n-1}} \le \left(e^{-\gamma} - 1\right) L_{12}^n. \tag{30}$$

Considering the accumulation of (30) according to (27) as n increases from 1 to N, we have the following inferences:

$$\ln \frac{A_N}{A_0} \le \left(e^{-\gamma} - 1\right) C_{L_{12}}^N. \tag{31}$$

By combining formulas (29) and (31), we achieve

$$C_{L_{12}}^N \le \frac{\gamma}{1 - e^{-\gamma}} C_{L_2}^N + \frac{\ln 2}{1 - e^{-\gamma}}.$$
 (32)

Let $\psi = \gamma/(1 - e^{-\gamma})$ and $\Phi = \ln 2/(1 - e^{-\gamma})$, then we rewrite (32) as $C_{L12}^N \leq \psi C_{L2}^N + \Phi$. With $\gamma = (1/\ln N)^{1/2}$, we have $\lim_{N\to\infty} \psi = 1$ following L'Hopital's Rule [63]. Moreover, considering the partial derivative of Φ with N, we have the inference that $\lim_{N\to\infty} (d\Phi/dN) \to 0$. Thus, Φ becomes a constant (i.e., 19.45 according to its curve) when $N \to \infty$. Finally, (32) can be reduced to

$$C_{L_{12}}^{N} \le \left[C_{L_{2}}^{N}\right]^{+} + \left[19.45\right]^{-} \le C_{L_{2}}^{N} + \operatorname{const}_{<19.45}.$$
 (33)

Hence, we obtain that $C_{L12}^N \leq C_{L1}^N + \text{const} < C_{L2}^N + \text{const}$ under the condition of setting $\gamma = (1/\ln N)^{1/2}$ when $C_{L1}^N > C_{L2}^N$.

Analogously, when $C_{L1}^N < C_{L2}^N$, the following result can be also achieved: $C_{L12}^N \le C_{L1}^N + \text{const} < C_{L2}^N + \text{const}$ under the condition of setting $\gamma = (1/\ln N)^{1/2}$. Hence, *Proposition* 1 holds. \Box

D. Algorithm Design and Analysis

Based on the above inferences, we design the algorithm of $L^{3}F$ as in Algorithm 1. As shown in Algorithm 1, its computational and storage costs are, respectively, $\Theta(N \times |R_{K}| \times f)$ and $\Theta(f \times \max\{|R_{K}|/f, |U|, |I|\})$, where the detailed complexity analyses are given in the Supplementary File of this paper. Note that both *N* and *f* are positive constants. Hence, an $L^{3}F$ model's computational and storage costs are both linear with $|R_{K}|$ and easy to resolve in real applications.

IV. EXPERIMENTS AND RESULTS

A. General Settings

1) Data Sets: Nine HiDS datasets are adopted in the experiments [17], [64], whose details are summarized in Table I.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination

IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS

Algorithm 1	1 L ³ F
-------------	--------------------

Input: R _K	
Operation	Cost
Initializing f , λ , η , $\alpha_1 = 0.5$, $\alpha_2 = 0.5$,	$\Theta(1)$
N = 1000	
Initializing P randomly	$\Theta(\mathbf{U} \times f)$
Initializing Q randomly	$\Theta(\mathbf{I} \times f)$
while $n \leq N$ && not converge	$\times N$
for each rating $y_{u,i}$ in R_K	$\times R_K $
for $d = 1$ to f	$\times f$
update $p_{u,d}^n$ according to (11)	$\Theta(1)$
update $q_{i,d}^n$ according to (11)	$\Theta(1)$
end for	
end for	
update α_1^n according to (15) and (19)	$\Theta(1)$
update α_2^n according to (15) and (19)	$\Theta(1)$
n = n + 1	$\Theta(1)$
end while	
Output: P, Q	

TABLE I Summary of Experimental Data Sets

No.	Name	U	I	$ R_K $	Density
D1	Dating [65]	135,359	168,791	17,359,346	0.08%
D2	Douban [17]	129,490	58,541	16,830,839	0.22%
D3	Eachmovie [3]	72,916	1,628	2,811,718	2.37%
D4	Epinion [65]	755,760	120,492	13,668,321	0.02%
D5	Flixter [66]	147,612	48,794	8,196,077	0.11%
D6	Jester [66]	24,983	100	1,186,324	47.49%
D7	MovieLens_10M [67]	71,567	65,133	10,000,054	0.21%
D8	MovieLens_20M [67]	138,493	26,744	20,000,263	0.54%
D9	[*] Yahoo-R2 [68]	200,000	136,736	76,344,627	0.28%

^{*}We select its first 200,000 users with more than 76 million ratings to build D9 for conducting the experiments.

2) Evaluation Metrics: In an RS, missing data prediction and ranking prediction (top-K recommendation) are commonly adopted to evaluate its performance [50]. Considering missing data prediction, mean absolute error (MAE) and root mean squared error (RMSE) are widely adopted as the evaluation metrics [9], [19]



where Γ denotes the testing set and $|\cdot|_{abs}$ calculates the absolute value of a given number. Note that a model's low MAE and RMSE indicate its high prediction accuracy.

On the other hand, the task of ranking prediction produces a ranked list with K items recommended to each user, where K is a cutoff parameter. Precision and normalized discounted

TABLE II SUMMARY OF COMPARED MODELS

Model	Description
L_1 - $LF_{\overline{b}}$	L_1 -LF is a basic LF model built on an L_1 -norm-oriented Loss. L_1 -LF _b stands for an L_1 -LF model without linear biases.
L_1 - LF_b	An L ₁ -LF model with linear biases.
L_2 - $LF_{\overline{b}}$	A sohposticated LF model proposed in 2009 [18]. It is built on an L_2 -norm-oriented <i>Loss</i> solely. L_2 -LF ₅ stands for an L_2 -LF model without linear biases
L ₂ -LF _b	An L ₂ -LF model with linear biases.
NLF	A sophisticated LF model proposed in 2016 [19]. It improves an L ₂ -LF model by introducing the non-negative constraints into it for efficiently describing non-negative data.
FNLF	A fast non-negative LF model based on a generalized mo- mentum method [9]. It is proposed in 2018 [9] and also relies on an L_2 -norm-oriented <i>Loss</i> solely.
AutoRec	A autoencoder-based model proposed in 2015 [51]. It is a representative DNN-based recommender model.
NRT	A DNN-based model proposed in 2017 [53]. It adopts a multi-task learning-incorporated framework based on multi-layered perceptron and recurrent neural networks.
DCCR	A DNN-based model proposed in 2019 [52]. It improves AutoRec with the consideration of two different networks.
$L^{3}F_{\overline{b}}$	A proposed L ³ F model without linear biases.
L ³ F _b	A proposed L ³ F model with linear biases.

cumulative gain (NDCG) is adopted to evaluate the ranking prediction accuracy [50], [69] of a model. The precision of a tested model is given as

Precision@K =
$$\frac{1}{|U|} \sum_{u \in U} \frac{|\Lambda_K(u)|}{K}$$

where $\Lambda_K(u)$ denotes the intersection between the top-*K* set generated by the model and the testing set by user *u*. NDCG of a tested model is given as

$$DCG@K(u, \hat{Y}) = \sum_{k=1}^{K} \frac{2^{\hat{y}_{u,k}} - 1}{\log_2(i+1)}$$
$$DCG@K(u, \Gamma) = \sum_{k=1}^{K} \frac{2^{y_{u,k}} - 1}{\log_2(i+1)}$$
$$NDCG@K = \frac{1}{|U|} \sum_{u \in U} \left(\frac{DCG@K(u, \hat{Y})}{DCG@K(u, \Gamma)} \right)$$

where $\hat{y}_{u,k}$ denotes the *k*th prediction in $\Lambda_K(u)$ in descending order and $y_{u,i}$ denotes the actual *k*th rating by user *u* in Γ in descending order. Note that both Precision and NDCG lie in the scale of [0, 1], where high Precision and NDCG of a tested model indicate its high ranking prediction accuracy.

3) Baselines: We compare $L^{3}F$ with nine related models with different characteristics, including six LF-based models $(L_1-LF_{\bar{b}}, L_1-LF_{\bar{b}}, L_2-LF_{\bar{b}}, L_2-LF_{\bar{b}}, NLF, and FNLF)$ and three DNN-based models (AutoRec, NRT, and DCCR). Note that we also test two versions of $L^{3}F$, i.e., $L^{3}F_{\bar{b}}$ (without linear bias) and $L^{3}F_{\bar{b}}$ (with linear bias). Table II gives their summary.

4) *Experimental Designs:* In the next experiments, we aim at answering the following research questions (RQs).

- 1) RQ.1. Will $L^{3}F$ achieve balanced aggregation effects with its L_{1} -and- L_{2} norm-oriented *Loss*?
- 2) RQ.2. How do outlier data affect $L^{3}F$?



Fig. 2. Changes of α_1 and α_2 of L^3F during the training process on D1. (a) $L^3F_{\overline{b}}$. (b) $L^3F_{\overline{b}}$.



Fig. 3. Training process of L^3F_{δ} , L_1 - LF_{δ} , and L_2 - LF_{δ} on D1. (a) MAE. (b) RMSE.

- 3) RQ.3. How do hyper-parameters affect $L^{3}F$?
- 4) RQ.4. Can $L^{3}F$ outperform its peers?

In detail, we adopt 80%-20% train-test settings. The training process of a tested model terminates if the number of consumed iterations reaches a preset threshold (i.e., 1000) or the error difference between two consecutive iterations is smaller than 10^{-6} . All experiments are run on a PC with 3.4-GHz i7 CPU and 64-GB RAM. The source code of this article is available on:

https://github.com/Wuziqiao/Resource-code.git.

B. $L^{3}F$'s Aggregation Effects (RQ.1)

In this section, we empirically study that how $L^{3}F$ achieves balanced aggregation effects between L_{1} and L_{2} norm-oriented *Losses* during its training process from two aspects, i.e., monitoring the changes of α_{1} and α_{2} and testing $L^{3}F$'s rating prediction accuracy.

1) Monitoring the Changes of α_1 and α_2 : The results on D1 are presented in Fig. 2. The complete results on all the datasets are recorded in Figs. S1 and S2 in the Supplementary File. From them, we find that during the training process of L^3F_b/L^3F_b , both α_1 and α_2 adaptively change first and then converge to the different constants on different datasets. This phenomenon indicates that L^3F adaptively controls the aggregation effects of L_1 and L_2 norm-oriented Losses according to the data characteristics during the training process.



Fig. 4. Outlier data sensitivity tests of $L^3F_{\bar{b}}$, L_1 - $LF_{\bar{b}}$, and L_2 - $LF_{\bar{b}}$ on the artificial dataset. (a) MAE. (b) RMSE.

MAE and RMSE. Fig. 3 presents the comparison results of models without linear biases on D1. The complete results on all the datasets are recorded in Figs. S3–S6 in the Supplementary File. Considering models without linear biases, from Figs. 3, S3, and S4, we see that: 1) when evaluated by MAE, $L^3F_{\bar{b}}$ and L_1 -LF_{\bar{b}} achieve very close results and they both outperform L_2 -LF_{\bar{b}} greatly and 2) when evaluated by RMSE, $L^3F_{\bar{b}}$ and L_2 -LF_{\bar{b}} greatly. Considering models with linear biases, we arrive at highly similar results from Figs S5 and S6 that L^3F_b and L_1 -LF_{\bar{b}} achieve much lower MAE than L_2 -LF_{\bar{b}} does, while L^3F_b and L_2 -LF_b does.

3) Summary: Based on the above results, we see that by selecting α_1 and α_2 with (15), L³F adaptively controls the aggregation effects of L_1 and L_2 norm-oriented *Losses* during its training process, making it achieve low MAE and RMSE simultaneously than LF models built on an L_1 or L_2 norm-oriented *Loss* solely. Hence, we conclude that L³F finely aggregates the effects of L_1 and L_2 norm-oriented *Losses* to achieve such balanced performance.

C. $L^{3}F$'s Robustness to Outlier Data (RQ.3)

1) Artificial Data Set: To test L^3F 's sensitivity to outlier data, we create an artificial HiDS matrix with the following characteristics: 1) it has 5000 users (rows) and 1000 items (columns) and 2) its known entries take 2% of the whole entry set only, which are generated at random in the range of [0, 1]. Afterward, outlier users who randomly pick a subset of items to assign them with the same maximum or minimum rating are gradually added into this dataset. The percentage of outlier users increases from 0% to 200% with an interval of 10% to test L^3F 's sensitivity to outlier data.

Note that the outlier users are added to the training set only, since in real applications we do not want a recommender to fit malicious users' attacks. Fig. S7 in the Supplementary File provides an example to further illustrate this set of experiments. The generated artificial dataset is available on https://github.com/Wuziqiao/Resource-code.git.

2) Analysis: We compare L^3F_b/L^3F_b with L_1-LF_b/L_1-LF_b and L_2-LF_b/L_2-LF_b . Fig. 4 depicts the comparison results of models without linear biases. Figs. S8 in the Supplementary File depicts the comparison results of models with linear biases. From them, we find that outlier data have different



Fig. 5. Prediction error of $L^3F_{\bar{b}}$ on D1 as f increases. (a) MAE. (b) RMSE.

impacts on compared models. In general, $L^{3}F_{b}/L^{3}F_{b}$ and L_{2} - LF_{b}/L_{2} - LF_{b} outperform L_{1} - LF_{b}/L_{1} - LF_{b} at initial steps. However, since L_{1} norm is less sensitive to outlier data than L_{2} norm, L_{2} - LF_{b}/L_{2} - LF_{b} has the lowest prediction accuracy while L_{1} - LF_{b}/L_{1} - LF_{b} has the highest when the percentage of outlier users grows over 100%.

Note that in several testing cases, $L^{3}F_{b}/L^{3}F_{b}$ outperforms L_1 - LF_b/L_1 - LF_b in MAE and L_2 - LF_b/L_2 - LF_b in RMSE. The reason is that $L^{3}F_{\bar{b}}/L^{3}F_{b}$ aggregates the merits of both L_1 and L_2 norm-oriented Losses with a carefully designed self-adaptive weighting strategy, which is consistent with the proof given in Section III-C. In detail, with few outlier users, $L^{3}F_{\bar{b}}/L^{3}F_{b}$ increases the weight of L_{2} norm, making it better describe such an HiDS matrix. Hence, L³F_b/L³F_b's MAE is lower than that of L₁-LF_b/L₁-LF_b. On the other hand, since L_1 norm is more robust to outlier data, $L^3F_{\rm b}/L^3F_{\rm b}$ increases the weight of L_1 norm as outlier data increase, thereby achieving lower RMSE than L_2 -LF_b/L₂-LF_b does. For example, MAE of $L^3F_{\overline{b}}/L^3F_{b}$ (0.1725/0.1718) is lower than that of L_1 -LF_b/L₁-LF_b (0.1762/0.1737) when there are no outlier users. RMSE of $L^{3}F_{\bar{b}}/L^{3}F_{b}$ (0.2328/0.2304) is also lower than that of L_2 -LF_b/L₂-LF_b (0.2466/0.2698) when the percentage of outlier users is 50%.

3) Summary: This set of experiments shows that $L^{3}F$ finely aggregates the merits of L_{1} and L_{2} norm-oriented Losses to well balance the model robustness (depending on L_{1} norm) and stability (depending on L_{2} norm). Hence, it achieves robustness to outlier data as well as precisely describes the known data of an HiDS matrix.

D. $L^{3}F$'s Sensitivity to Its Hyper-Parameters (RQ.3)

In this section, we analyze L^3F 's behaviors with f, λ , and η . 1) With f: According to prior research [18], a large fenables an LF model to better describe an HiDS matrix unless f increases over its actual rank, which is reflexed by the increasing prediction accuracy of an LF model as f increases. Considering L^3F_5/L^3F_b , this conclusion is also true. Fig. 5 shows the prediction error of L^3F_5 on D1 as f increases. The complete results of L^3F_5/L^3F_b on all the datasets are drawn in Figs. S9–S12 in the Supplementary File. From these results, we see that in most testing cases, L^3F_5/L^3F_b 's RMSE and MAE decrease as f increases (besides the testing cases on D4 where L^3F_5/L^3F_b 's MAE increases as f increases). However, this accuracy gain becomes less significant as f increases over a certain threshold, e.g., 20



Fig. 6. Performance of $L^3F_{\tilde{b}}$ with respect to λ and η on D1. (a) MAE. (b) RMSE.

on most testing cases. Besides, according to Section III-D, we see that L^3F_b/L^3F_b 's time cost increases linearly with f. Hence, f should be appropriately chosen to carefully balance prediction accuracy and computational cost. Commonly, it is chosen from the scale of [10], [18]–[20], [34].

2) With λ and η : In this set of experiments, we perform a grid-based search to test the performance of L^3F_b/L^3F_b when λ increases from 0.01 to 0.1 and η increases from 0.0001 to 0.01. Fig. 6 presents the results of L^3F_b on D1. The complete results of L^3F_b/L^3F_b on all the datasets are recorded in Figs. S13–S16 in the Supplementary File. From them, we see the following.

- 1) Both λ and η affect the rating prediction accuracy of $L^{3}F_{\bar{b}}/L^{3}F_{b}$. As λ and η increase, MAE/RMSE decreases till λ and η reach their optimal thresholds, and then increase again. For example, on D8, the RMSE of $L^{3}F_{\bar{b}}$ decreases from 0.7866 to 0.7745 initially, and then it increases to 0.8285 as λ and η keep increasing.
- Generally speaking, η should be set relatively small to well balance the prediction accuracy and convergence rate, and λ should be tuned carefully since it is domain-specific [18], [31]. Similar results are also achieved by L³F_b/L³F_b: On different datasets, optimal values of η are close while optimal values of λ are different. More specifically, the optimal η is around 0.001 on each dataset, while the optimal λ lies in the range of [0.02, 0.09] on D1-9 for L³F_b/L³F_b.

3) Summary: We see that $L^{3}F$'s performance is closely connected with f, λ , and η . Empirically, it is suggested to set f = 10-20 and $\eta = 0.001$ in real applications. For λ , it should be carefully tuned on the target dataset.

E. Comparison Between $L^{3}F$ and Baselines (RQ.4)

In the comparisons, we evaluate rating prediction accuracy, ranking prediction accuracy, and computational efficiency. To draw fair comparisons, we adopt the following settings: 1) setting f = 20 for all the LF-based models; 2) adopting five-fold cross-validations and report the average results; 3) tuning the other hyper-parameters on one fold of each dataset to achieve the best performance of each model and then adopting the same values on the remaining fourfold; 4) adopting same random initialization method for the initial solution of each model; 5) adopting SGD to optimize each model with same order of training samples; 6) on rating prediction comparison, respectively, tuning hyper-parameters for achieving lowest MAE and RSME; and 7) on ranking prediction comparison, hyper-parameters are the same as that set in rating prediction comparison with lowest RMSE.

1) Comparison of Rating Prediction Accuracy: Table III presents the detailed comparison results, where we see that $L^3F_{\bar{b}}/L^3F_b$ and $L_1-LF_{\bar{b}}/L_1-LF_b$ have lower MAE than the other models. On RMSE, $L^3F_{\bar{b}}/L^3F_b$, $L_2-LF_{\bar{b}}/L_2-LF_b$, AutoRec, and DCCR achieve better performance than the other models. In particular, we see that $L^3F_{\bar{b}}/L^3F_b$ outperforms $L_1-LF_{\bar{b}}/L_1-LF_b$ in MAE and $L_2-LF_{\bar{b}}/L_2-LF_b$ in RMSE on some datasets. The reason is same as analyzed in Section IV-C, i.e., $L^3F_{\bar{b}}/L^3F_b$ aggregates the merits of both robustness (depending on L_1 norm) and stability (depending on L_2 norm) to well describe an HiDS matrix with different situations of outliers. By comparing $L^3F_{\bar{b}}$ and L^3F_b , we find that linear bias can improve L^3F 's rating prediction accuracy on several cases but not on all. For example, RMSE is reduced from 0.5969 to 0.4775 on D4 while increased from 1.0601 to 1.0776 on D9.

To check whether $L^{3}F_{\bar{b}}/L^{3}F_{b}$ has a higher rating prediction accuracy than the other models, we conduct statistical analysis on Table III. First, the win/tie/loss counts of $L^{3}F_{\bar{b}}/L^{3}F_{b}$ versus other models one by one are summarized in the third/second-to-last row of Table III, which indicates that $L^{3}F_{b}/L^{3}F_{b}$ achieves a higher rating prediction accuracy than other models on most datasets. Second, we perform Friedman test [70] because it is an effective statistical test method in validating the performance of multiple models on multiple datasets. Friedman test results on the MAE/RMSE of Table III are recorded in the last row of Table III, where it accepts the hypothesis that these comparison models have significant differences with a significance level of 0.05. Friedman test results show that $L^{3}F_{b}/L^{3}F_{b}$ has a higher rating prediction accuracy than the other models. Further, L³F_b has a smaller F-rank value than $L^{3}F_{\bar{b}}$, which means that $L^{3}F_{b}$ outperforms $L^{3}F_{\bar{b}}$ in general.

Besides, to check whether $L^{3}F_{b}$ achieves significantly higher rating prediction accuracy than the other models, we conduct the Wilcoxon signed-ranks test [71], [72] on the results of Table III. In detail, we compare MAE/RMSE of $L^{3}F_{b}$ with that of the other models one by one. Wilcoxon signed-ranks test is a nonparametric pairwise comparison procedure and has three indicators—R+, R-, and p-value. The larger R+ value indicates higher performance and the p-value indicates the significance level. Table S.II in the Supplementary File records the test results, where we see that $L^{3}F_{b}$ has a significantly higher rating prediction accuracy than all the comparison models with a significance level of 0.05 except for $L^{3}F_{\bar{b}}$. However, we see that $L^{3}F_{b}$ achieves a larger R+ when comparing with $L^{3}F_{\bar{b}}$, which means that linear bias can slightly boost L³F's rating prediction accuracy in general.

2) Comparison on Ranking Prediction Accuracy: We consider two situations. First is to check whether the models can correctly rank the items of testing set. Second is to simulate a real recommendation scenario. *Situation* 1: predicting ratings for each item of testing set first and then calculating NDCG. Note that the Precision of *Situation* 1 is 1 because all the items



Fig. 7. Comparison CPU running time of involved models on D1-D9.

of testing set are predicted and ranked. *Situation* **2**: predicting ratings for the items that are not rated by the users of training set first and then calculating Precision and NDCG. Since it is extremely time-consuming to predict and rank all items under *Situation* **2**, we followed the common strategy [46] that randomly samples 500 items.

Situation 1: Table IV presents the detailed comparison results. Meanwhile, we also conduct two statistical analyses on these comparison results, i.e., win/tie/loss counts of $L^3F_{\bar{b}}/L^3F_b$ versus other models and Friedman test among all the models. From Table IV, we find that: 1) $L^3F_{\bar{b}}$ and L_2 -LF_{\bar{b}} achieve a much higher NDCG than the other models; 2) $L^3F_{\bar{b}}$ has a slightly better performance than L_2 -LF_{\bar{b}}; and 3) $L^3F_{\bar{b}}$ performs better than L^3F_b .

Situation 2: Tables V and VI present the detailed comparison results, where win/tie/loss counts and Friedman test results are also recorded. From them, we observe that: 1) $L^{3}F_{\bar{b}}$ and L_{1} -LF_{\bar{b}} achieve a much higher Precision and NDCG than the other models and 2) $L^{3}F_{\bar{b}}$ has a slightly better performance than L_{1} -LF_{\bar{b}}, and c) $L^{3}F_{\bar{b}}$ performs better than $L^{3}F_{b}$.

Analysis: Under Situation 1, $L^{3}F_{\bar{b}}$ and L_{2} -LF_b perform better than the other models, which coincide with the result obtained from rating prediction comparison. While under Situation 2, $L^{3}F_{\bar{b}}$ and L_{1} -LF_b do better. One reason is that both rating prediction comparison and Situation 1 are tested on each item of testing set, where L_2 norm-oriented Loss can search for a better solution. Under Situation 2, only a small part of randomly sampled 500 items is included in the testing set. These randomly sampled items that are not included in the testing set maybe like outliers in training a model. As a result, L_1 norm-oriented Loss can search for a better solution. Since $L^{3}F_{\bar{b}}$ has the merits of both L_{1} and L_{2} norm-oriented Loss, it performs well under both situations. Note that $L^{3}F_{\bar{b}}$ outperforms $L^{3}F_{b}$ under both situations, which means that linear bias tends to degrade instead of improving L³F's ranking prediction accuracy.

3) Comparison of Computational Efficiency: Fig. 7 presents the CPU running time of all the models when training for rating prediction, where we observe that: 1) DNN-based models (AutoRec, NRT, and DCCR) cost much more CPU running time than the other models, which is caused by their DNN-based learning strategy [58]; 2) $L^{3}F$ costs a little more CPU running time than L_1 -LF and L_2 -LF because it

IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS

TABLE III

Dataset	Metric	L_1 - $LF_{\overline{b}}$	L_1 - LF_b	L_2 - $LF_{\overline{b}}$	L_2 - LF_b	NLF	FNLF	AutoRec	NRT	DCCR	$L^{3}F_{\overline{b}}$	$L^{3}F_{b}$
D1	MAE	1.1606	1.1492	1.23920•	1.23280•	1.2617○●	1.2588○●	1.2610○●	1.2303○●	1.25740•	1.1740	1.1781
	RMSE	1.8672○●	1.8025○●	1.8066○●	1.7809•	1.8245○●	1.8215○●	1.8027○●	1.8101○●	1.8013○●	1.7987	1.7704
D2	MAE	0.5394•	0.5245	0.5537○●	0.5516○●	0.5590○●	0.5592○●	0.5606○●	0.5675○●	0.55810•	0.5412	0.5347
D2	RMSE	0.7344○●	0.7222○●	0.7139•	0.6993	0.7150○●	0.7139•	0.7080•	0.7106○●	0.7074•	0.7145	0.7000
D	MAE	0.1713○●	0.1705○●	0.1732○●	0.1730○●	0.1767○●	0.1763○●	0.1784○●	0.1774○●	0.1775○●	0.1700	0.1698
D3	RMSE	0.22900•	0.2278○●	0.22510	0.2262○●	0.2264○●	0.2259○●	0.2305○●	0.23010•	0.2289○●	0.2249	0.2256
D4	MAE	0.2517○●	0.2223	0.3011○●	0.2938○●	0.3047○●	0.30360•	0.3014○●	0.3002○●	0.30360•	0.2512	0.2422
D4	RMSE	0.6100○●	0.4875●	0.5958•	0.4821•	0.5994○●	0.5977○●	0.5946•	0.5926•	0.5952•	0.5969	0.4775
DC	MAE	0.63190•	0.6042	0.6447○●	0.6207•	0.6550○●	0.6520○●	0.6295•	0.63370•	0.6308•	0.6318	0.6050
D5	RMSE	0.9098○●	0.8450•	0.8961○●	0.8383•	0.9056○●	0.9038○●	0.8682•	0.8677●	0.8792•	0.8960	0.8326
D	MAE	0.7580○●	0.7600○●	0.7664○●	0.7676○●	0.7769○●	0.7778○●	0.7905○●	0.7878○●	0.7883○●	0.7562	0.7578
D6	RMSE	1.0206○●	1.0177○●	0.9957•	0.9982○●	1.0049○●	1.0003○●	1.0078○●	1.0056○●	1.0042○●	0.9978	0.9935
D7	MAE	0.59860•	0.6038○●	0.5999○●	0.6067○●	0.6080○●	0.6068○●	0.6048○●	0.6035○●	0.6002○●	0.5956	0.5981
D/	RMSE	0.7953○●	0.8048○●	0.7819	0.7936○●	0.78930	0.78810	0.78650	0.78340	0.78470	0.7821	0.7899
Dû	MAE	0.5877○●	0.5927○●	0.5886○●	0.5955○●	0.5977○●	0.5961○●	0.5947○●	0.60180•	0.5902○●	0.5848	0.5863
D8	RMSE	0.7873○●	0.7960○●	0.7737	0.7848○●	0.7819○●	$0.7798 \circ$	$0.7802 \circ$	0.77980	0.77890	0.7743	0.7806
DO	MAE	0.7661	0.7916○●	0.8037○●	0.8475○●	0.8342○●	0.8228○●	0.8029○●	0.8221○●	0.8058○●	0.7742	0.7840
D9	RMSE	1.1129○●	1.1460○●	1.0596	1.0968○●	1.07460	$1.0704 \circ$	1.06290	1.1150○●	1.0931○●	1.0601	1.0776
Q. 1. 1. 1	owin/tie/loss	15/0/3	12/0/6	12/0/6	13/0/5	18/0/0	17/0/1	14/0/4	16/0/2	14/0/4	_	
Analysia	•win/tie/loss	16/0/2	14/0/4	14/0/4	17/0/1	16/0/2	15/0/3	15/0/3	16/0/2	16/0/2		
Analysis	F-rank*	6.83	5.50	4.97	5.56	8.83	7.64	7.11	7.19	6.31	3.28	2.78

Comparison Results on Rating Prediction Accuracy, Including Win/Tie/Loss Counts Statistic and Friedman Test, Where \circ and \bullet Indicate That the Rating Prediction Accuracy of L^3F_5 and L^3F_b Is Higher Than or Same to That of Comparison Models, Respectively

* The smaller F-rank value denotes a higher rating prediction accuracy.

TABLE IV

 $\label{eq:comparison} Comparison Results on NDCG Under Situation 1, Including Win/Tie/Loss Counts Statistic and Friedman Test, Where \circ and \bullet Indicate that the NDCG of L^3F_b and L^3F_b Is Higher Than or Same to That of Comparison Models, Respectively $$\circ$ Nodels, Respectively $$\circ$ Nodels, Respectively $$\circ$ Nodels, Respectively $$\circ$ Nodels, $$\circ$$

Dataset	Metric	L_1 - $LF_{\overline{b}}$	L_1 - LF_b	L_2 - $LF_{\overline{b}}$	L_2 - LF_b	NLF	FNLF	AutoRec	NRT	DCCR	$L^{3}F_{\overline{b}}$	L ³ F _b
DI	NDCG@5	0.90300•	0.90580•	0.9056○●	0.9040○●	0.9020○●	0.9025○●	0.8963○●	0.8951○●	0.89820•	0.9068	0.9061
DI	NDCG @10	0.9110○●	0.9137○●	0.9137○●	0.9123○●	0.9106○●	0.9109○●	0.9052○●	0.9061○●	0.9092○●	0.9148	0.9142
D2	NDCG@5	0.8588○●	0.8600○●	0.8701	0.8703	0.8647○●	0.8643○●	0.8608○●	0.8565○●	0.8631○●	0.8696	0.8700
D_{2}	NDCG @10	0.8777○●	0.8794○●	0.8869○●	0.8879	0.8817○●	0.8822○●	0.8855○●	0.8758○●	0.8862○●	0.8874	0.8877
D2	NDCG@5	0.9060○●	0.9062○●	0.9076○●	0.9059○●	0.9088	0.9090	0.9064○●	0.9071○●	0.9081	0.9077	0.9077
D3	NDCG @10	0.9254○●	0.9251○●	0.9267•	0.9253○●	0.9275	0.9276	0.9273	0.9288	0.9286	0.9268	0.9262
D4	NDCG@5	0.97870	0.9786○●	0.9807	0.97950	0.9808	0.9809	0.9731○●	0.9783○●	0.9723○●	0.9800	0.9786
D4	NDCG @10	0.98250	0.9821○●	0.98350	$0.9828 \circ$	0.9850	0.9856	0.9776○●	0.9819○●	0.9762○●	0.9835	0.9821
DC	NDCG@5	0.8970○●	0.8977○●	0.8985○●	0.8982○●	0.8974○●	0.8976○●	0.8770○●	0.8820○●	0.8792○●	0.8986	0.8993
DS	NDCG @10	0.91130•	0.9119○●	0.9124○●	0.9125○●	0.9114○●	0.9116○●	0.8934○●	0.8980○●	0.8952○●	0.9126	0.9132
D6	NDCG@5	0.7431○●	0.7433○●	0.7489•	0.7447○●	0.7483○●	0.7482○●	0.7294○●	0.7288○●	0.7311○●	0.7483	0.7494
D0	NDCG @10	0.81800•	0.8177○●	0.8217•	0.8204○●	0.8208○●	0.8211○●	0.8071○●	0.7970○●	0.8096○●	0.8214	0.8223
D7	NDCG@5	0.82470	0.8202○●	0.82980	0.8231○●	$0.8288 \circ$	0.82900	0.8167○●	0.8020○●	0.8181○●	0.8299	0.8245
\mathbf{D}	NDCG @10	0.85860•	0.8550○●	$0.8627 \circ$	0.8577○●	0.86150	0.86180	0.8444○●	0.83480•	0.8502○●	0.8627	0.8588
D٥	NDCG@5	0.82570	0.8213○●	0.8299	0.8235○●	$0.8287 \circ$	0.82900	0.8110○●	0.8117○●	0.81840•	0.8297	0.8253
D8	NDCG @10	0.85960	0.8561○●	0.86311	0.8582○●	0.86190	0.86230	0.8472○●	0.8427○●	0.8487○●	0.8630	0.8593
D0	NDCG@5	0.80200	0.7890○●	0.8193	0.80460	0.81030	0.81160	0.7785○●	0.7748○●	0.7815○●	0.8154	0.8016
D9	NDCG @10	0.82170	0.8106○●	0.83380	0.82430	0.82930	0.82990	0.7985○●	0.7997○●	0.8005○●	0.8339	0.8214
Statistical	owin/tie/loss	18/0/0	18/0/0	9/2/7	16/0/2	13/1/4	14/0/4	17/0/1	17/0/1	16/0/2		
Analysis	•win/tie/loss	11/0/7	16/2/0	9/0/9	12/0/6	8/0/10	8/0/10	17/0/1	17/0/1	16/0/2	_	
Analysis	F-rank*	5.06	4.75	9.14	6.44	7.31	8.06	1.72	2.67	3.72	9.44	7.69

* The larger F-rank value denotes better performance on NDCG.

ensembles L_1 and L_2 norms; and 3) L³F cost less or more CPU running time than NLF and FNLF on the different datasets.

Note that vanilla SGD-based matrix factorization for RSs can be efficiently computed in parallel [73]. Since L³F also belongs to the family of SGD-based matrix factorization,

we can improve its computational efficiency through parallelization. On this basis, we develop $L^{3}F$ to a parallel version according to Hogwild! [73]. Specifically, we randomly sample the known ratings from R_{K} first and then employ them to, respectively, update $L^{3}F$'s each LF through different threads simultaneously. Please refer to [73] for details.

TABLE V

 $\label{eq:comparison} \begin{array}{l} \text{Comparison Results on Precision Under Situation 2, Including Win/Tie/Loss Counts Statistic and Friedman Test, Where \circ and \circ Indicate That the Precision of L^3F_b and L^3F_b Is Higher Than or Same to That of Comparison Models, Respectively $=$ Note: L^3F_b and $L^$

Dataset	Metric	L_1 - $LF_{\overline{b}}$	L_1 - LF_b	L_2 - $LF_{\overline{b}}$	L ₂ -LF _b	NLF	FNLF	AutoRec	NRT	DCCR	$L^{3}F_{\overline{b}}$	$L^{3}F_{b}$
D1	Precision@5	0.0044	0.00100•	0.00300	0.00110	0.00190	0.00180	0.00190	0.00270	0.00220	0.0042	0.0010
DI	Precision@10	0.0031	0.0007○●	$0.0022 \circ$	0.0007○●	0.00130	0.00120	0.00160	0.00210	$0.0017 \circ$	0.0029	0.0007
D2	Precision@5	0.0217	0.0038○●	0.01870	0.0038○●	0.00940	0.01020	0.00480	0.00730	0.00460	0.0216	0.0038
D_{2}	Precision@10	0.0155	0.0033○●	0.01340	0.00340•	$0.0079 \circ$	$0.0082\circ$	0.0033○●	0.00630	0.0031○●	0.0153	0.0034
D3	Precision@5	0.14930	0.0273○●	0.15260	0.0271○●	$0.0881\circ$	0.09740	0.04920	$0.0677 \circ$	0.05230	0.1544	0.0273
05	Precision@10	0.12200	0.02510	0.12570	0.0220○●	0.0779 \circ	0.08450	0.03100	$0.0587\circ$	0.03510	0.1260	0.0221
D4	Precision@5	0.00360	0.0018○●	0.00360	0.0018○●	0.0018○●	0.0016○●	0.0011○●	0.0015○●	0.0013○●	0.0036	0.0018
D4	Precision@10	0.00310	0.0018○●	0.00310	0.0018○●	0.0018○●	0.0017○●	0.0010○●	0.00120•	0.0011○●	0.0031	0.0018
D5	Precision@5	0.02420	$0.0057 \circ$	0.02240	$0.0057 \circ$	0.01160	0.01120	$0.0071 \circ$	0.01860	0.00680	0.0247	0.0055
D3	Precision@10	0.01770	0.0055○●	$0.0174 \circ$	0.0055○●	0.01060	$0.0102 \circ$	$0.0067 \circ$	$0.0148 \circ$	$0.0065 \circ$	0.0179	0.0057
D6	Precision@5	0.95680	0.69570	$0.9272 \circ$	0.68120•	$0.7270 \circ$	0.71500	0.81720	0.87320	0.82410	0.9577	0.6906
D0	Precision@10	0.8824	0.59250	0.76830	0.5778○●	0.62170	0.64500	0.72830	0.77260	0.73690	0.8181	0.5812
D7	Precision@5	0.02510	$0.0105\circ$	0.02380	0.0099○●	0.0099○●	$0.0104 \circ$	0.0084○●	0.01180	0.0092○●	0.0254	0.0099
D	Precision@10	0.01670	$0.0084\circ$	0.01630	0.0074○●	0.0065○●	0.0067○●	0.0055○●	0.00790	0.0061○●	0.0168	0.0074
D8	Precision@5	$0.0403 \circ$	$0.0187 \circ$	0.03590	$0.0187 \circ$	0.01180•	0.0122○●	0.0075○●	0.0174○●	0.00930•	0.0408	0.0176
D0	Precision@10	0.02890	0.01370	0.02640	0.0129○●	0.0084○●	0.0096○●	0.0066○●	0.01240•	0.0075○●	0.0291	0.0130
D9	Precision@5	0.0101	0.00060	$0.0092\circ$	0.00060	0.00140	$0.0015 \circ$	$0.0054\circ$	$0.0067\circ$	$0.0058\circ$	0.0100	0.0005
D9	Precision@10	$0.0077\circ$	$0.0007\circ$	0.00680	0.0006○●	0.00110	0.00110	0.00440	0.00580	0.00420	0.0077	0.0006
Statistical	owin/tie/loss	9/3/6	18/0/0	16/2/0	18/0/0	18/0/0	18/0/0	18/0/0	18/0/0	18/0/0		—
Analysis	•win/tie/loss	0/0/18	2/6/10	0/0/18	6/8/4	3/3/12	5/0/13	7/0/11	4/0/14	7/0/11	—	—
Anarysis	F-rank*	10.31	4.14	8.22	3.42	5.44	5.47	3.89	6.78	4.33	10.53	3.47

* The larger F-rank value denotes better performance on Precision.

TABLE VI

 $\begin{array}{l} \text{Comparison Results on NDCG Under Situation 2, Including Win/Tie/Loss Counts Statistic and Friedman Test, Where \circ and \bullet Indicate That the NDCG of L^3F_b and L^3F_b Is Higher Than or Same to That of Comparison Models, Respectively \circ Note: That the NDCG of L^3F_b and L^3F_b is the same to that the NDCG of L^3F_b and L^3F_b is the same to the note: L^3F_b and L^3F_b is the same to the note: L^3F_b and L^3F_b is the same to the note: L^3F_b and L^3F_b is the same to the note: L^3F_b and L^3F_b is the note: L^3F_b and L^3F_b is the same to the note: L^3F_b and L^3F_b and L^3F_b is the note: L^3F_b and L^3F_b is the note: L^3F_b and L^3F_b and$

Dataset	Metric	L_1 - $LF_{\overline{b}}$	L_1 - LF_b	L_2 - $LF_{\overline{b}}$	L_2 - LF_b	NLF	FNLF	AutoRec	NRT	DCCR	$L^{3}F_{\overline{b}}$	$L^{3}F_{b}$
DI	NDCG@5	0.00330	0.0019○●	0.00300	0.0019○●	0.0016○●	0.0015○●	0.0017○●	0.00230	0.0019○●	0.0035	0.0019
DI	NDCG @10	0.00320	0.0018○●	$0.0029\circ$	0.0018○●	0.0013○●	0.0010○●	$0.0020\circ$	0.00190	0.00210	0.0033	0.0018
D2	NDCG@5	0.0203	$0.0048 \circ$	0.01830	$0.0048 \circ$	0.00830	0.00930	0.0042○●	$0.0085\circ$	0.0040○●	0.0200	0.0047
D_{2}	NDCG @10	0.0188	0.0054○●	0.01730	0.0054○●	$0.0076 \circ$	$0.0088\circ$	0.0030○●	0.00960	0.0028○●	0.0180	0.0055
D2	NDCG@5	0.15860	0.0379○●	0.16440	0.0378○●	0.09530	$0.1088 \circ$	0.05140	0.05850	0.05470	0.1662	0.0380
05	NDCG @10	0.15690	$0.0442\circ$	0.16380	0.0389○●	0.09170	0.09740	0.0342○●	0.06960	0.0371○●	0.1639	0.0392
D4	NDCG@5	$0.0028 \circ$	0.0001○●	0.00280	0.0001○●	0.00130	0.00120	$0.0009\circ$	0.00130	0.00110	0.0028	0.0001
D4	NDCG @10	$0.0025 \circ$	0.0001○●	0.00240	0.0002○●	$0.0012 \circ$	0.00110	$0.0009 \circ$	0.00110	$0.0010\circ$	0.0025	0.0002
Df	NDCG@5	0.01640	0.0013○●	0.01490	0.0013○●	$0.0088 \circ$	$0.0083 \circ$	$0.0075 \circ$	0.01190	0.00710	0.0167	0.0014
D3	NDCG @10	0.01420	0.0013○●	0.01390	0.00140	0.00760	$0.0078 \circ$	$0.0070\circ$	0.01130	$0.0068 \circ$	0.0142	0.0013
D6	NDCG@5	0.6711	0.56100	0.66350	0.5495○●	0.7449	0.7396	0.7148	0.7196	0.7264	0.6704	0.5581
D0	NDCG @10	0.69040	0.58630	0.67980	0.5786○●	0.8199	0.8220	0.7427	0.7620	0.7592	0.6923	0.5804
D7	NDCG@5	0.02120	0.01150	0.02130	0.0109○●	0.0099○●	0.0102○●	0.0094○●	0.01280	0.0099○●	0.0217	0.0109
\mathbf{D}_{i}	NDCG @10	0.01900	0.01210	0.01910	$0.0107 \circ$	0.0075○●	0.0087○●	0.0085○●	0.01200	0.0087○●	0.0195	0.0105
D٥	NDCG@5	0.03610	0.02140	0.03180	0.02150	0.0115○●	0.0118○●	0.0063○●	0.0198○●	0.0075○●	0.0375	0.0202
D8	NDCG @10	0.03360	0.02060	0.03120	0.0196○●	0.0180○●	0.0182○●	0.0056○●	0.01990	0.0063○●	0.0352	0.0196
D0	NDCG@5	0.0089	0.0007○●	$0.0078\circ$	0.0007○●	0.00140	0.00160	0.00660	$0.0072\circ$	0.00690	0.0088	0.0007
D9	NDCG @10	$0.0088 \circ$	$0.0011 \circ$	$0.0075 \circ$	0.0009○●	$0.0017 \circ$	0.00180	$0.0067 \circ$	$0.0071 \circ$	0.00660	0.0088	0.0009
Statistical	owin/tie/loss	10/4/4	18/0/0	17/1/0	18/0/0	16/0/2	16/0/2	16/0/2	16/0/2	16/0/2	_	_
Analysis	•win/tie/loss	0/0/18	4/5/9	0/0/18	6/8/4	6/0/12	6/0/12	8/0/10	1/0/17	7/1/10	—	—
Analysis	F-rank*	9.53	3.97	8.72	3.39	5.39	5.83	3.83	7.28	4.47	10.08	3.50

* The larger F-rank value denotes better performance on NDCG.

Next, we test the computational efficiency of parallel L^3F with different numbers of threads. Fig. 8 records the results on D1. The complete results on all datasets are recorded in Fig. S17 in the Supplementary File. From them, we see that L^3F 's computational efficiency has been significantly improved with a nearly linear speedup as the number of threads increases. For example, the CPU running time with

16 threads (137.04 s) is much less than that with only one thread (1284.69 s, original $L^{3}F$ without parallelization) on D1. Note that there are no significant differences in prediction accuracy between original $L^{3}F$ and its parallel version.

4) Summary of Comparison Results: We compare $L^3F_{\bar{b}}/L^3F_b$ with nine related state-of-the-art models on rating prediction, ranking prediction, and computational



Fig. 8. Computational efficiency of parallel $L^{3}F$ on D1 as the number of threads increases. (a) CPU running time. (b) Speedup.

efficiency. The comparison results verify that: 1) $L^{3}F_{b}$ has a significantly higher rating prediction accuracy than the nine models; 2) $L^{3}F_{\bar{b}}$ performs best on both situations of ranking prediction among all the models; 3) linear bias has positive effects in improving $L^{3}F$'s rating prediction accuracy, while it can hardly improve $L^{3}F$'s ranking optimization performance; 4) $L^{3}F$'s computational efficiency is higher than that of DNN-based models and comparable to that of LF-based models; and 5) $L^{3}F$'s computational efficiency can be significantly improved through parallelization.

According to recent LF analysis on an HiDS matrix, a linear LF model like L_2 -LF_b/ L_2 -LF_b may outperform DNNbased models with careful model settings. In this study, each involved model's hyper parameters are carefully tuned on D1–D9 to achieve its best performance. Therefore, we find that L_1 -LF_b/ L_1 -LF_b, L_2 -LF_b/ L_2 -LF_b, NLF, and FNLF outperform AutoRec, NRT, and DCCR in terms of prediction accuracy for missing data of an HiDS matrix on several testing cases. This phenomenon is also highly consistent with the study presented in [59] and [60]. On the other hand, this accuracy gain is also data-dependent since the involved DNN-based models can outperform these linear models on the other testing cases. However, a proposed L^3F outperforms its peers in terms of prediction accuracy for missing data on most testing cases.

Note that a DNN-based model costs much time to build, while an LF model enjoys its high computational efficiency on HiDS data. From this point of view, a proposed L³F model can better satisfy the demands of industrial RSs for fast and accurate recommendations than its peers do.

V. CONCLUSION

This study proposes an $L^{3}F$ model for efficiently addressing HiDS matrices arising from RSs. It adopts twofold ideas: 1) Aggregating the robustness depending on L_1 norm and stability depending on L_2 norm to form its L_1 -and- L_2 -normoriented *Loss* and 2) Adaptively adjusting weights of L_1 and L_2 norms in its *Loss*. Theoretical and empirical studies show that an $L^{3}F$ model obtains fine aggregation effects with L_1 norm-oriented *Loss*'s robustness and L_2 norm-oriented *Loss*'s stability when handling an HiDS matrix with outliers.

According to Section IV-D, an L³F model's performance is sensitive to its regularization coefficient λ , which is data-dependent and should be tuned carefully. Therefore, its self-adaptation is desired to enhance L³F's practicability. On the other hand, will L³F achieve further performance gain with the incorporation of more distance metrics into its *Loss*, e.g., $L_{2,1}$ norm? Naturally, more complex weighting strategies are needed to do so. We plan to address these open issues in the future.

ACKNOWLEDGMENT

The authors highly appreciate Dr. Yi He working at the University of Louisiana at Lafayette, Lafayette, USA, for his efforts in the theoretical studies presented in Section III-C, and Jiajia Jiang for his efforts in the empirical studies of this article.

REFERENCES

- X. Wu, X. Zhu, G.-Q. Wu, and W. Ding, "Data mining with big data," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 97–107, Jan. 2014.
- [2] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Inf. Fusion*, vol. 42, pp. 146–157, Jul. 2018.
- [3] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges," *ACM Comput. Surv.*, vol. 47, no. 1, pp. 1–45, 2014.
- [4] R. Lu, X. Jin, S. Zhang, M. Qiu, and X. Wu, "A study on big knowledge and its engineering issues," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 9, pp. 1630–1644, Sep. 2019.
- [5] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2015, pp. 77–118, doi: 10.1007/978-1-4899-7637-6_3.
- [6] D. Liang, R. G. Krishnan, M. D. Hoffman, and T. Jebara, "Variational autoencoders for collaborative filtering," in *Proc. World Wide Web Conf. World Wide Web (WWW)*, 2018, pp. 689–698.
- [7] A. Liu *et al.*, "Differential private collaborative Web services QoS prediction," *World Wide Web*, vol. 22, pp. 2697–2720, Apr. 2018, doi: 10.1007/s11280-018-0544-7.
- [8] D. Ryu, K. Lee, and J. Baik, "Location-based Web service QoS prediction via preference propagation to address cold start problem," *IEEE Trans. Services Comput.*, early access, Apr. 2, 2018, doi: 10.1109/TSC.2018.2821686.
- [9] X. Luo, Z. Liu, S. Li, M. Shang, and Z. Wang, "A fast non-negative latent factor model based on generalized momentum method," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 610–620, Jan. 2021.
- [10] L. Wu, P. Sun, R. Hong, Y. Ge, and M. Wang, "Collaborative neural social recommendation," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 464–476, Jan. 2021.
- [11] J.-D. Zhang, C.-Y. Chow, and J. Xu, "Enabling kernel-based attributeaware matrix factorization for rating prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 4, pp. 798–812, Apr. 2017.
- [12] J. Castro, J. Lu, G. Zhang, Y. Dong, and L. Martinez, "Opinion dynamics-based group recommender systems," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 48, no. 12, pp. 2394–2406, Dec. 2018.
- [13] W. Luan, G. Liu, C. Jiang, and L. Qi, "Partition-based collaborative tensor factorization for POI recommendation," *IEEE/CAA J. Automatica Sinica*, vol. 4, no. 3, pp. 437–446, Jul. 2017.
- [14] M. Gong, X. Jiang, H. Li, and K. C. Tan, "Multiobjective sparse nonnegative matrix factorization," *IEEE Trans. Cybern.*, vol. 49, no. 8, pp. 2941–2954, Aug. 2019.
- [15] X. He, J. Tang, X. Du, R. Hong, T. Ren, and T. Chua, "Fast matrix factorization with nonuniform weights on missing data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 8, pp. 2791–2804, Aug. 2020.
- [16] Z. Li, J. Tang, and X. He, "Robust structured nonnegative matrix factorization for image representation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1947–1960, May 2018.
- [17] X. Luo, Z. Wang, and M. Shang, "An instance-frequency-weighted regularization scheme for non-negative latent factor analysis on highdimensional and sparse data," *IEEE Trans. Syst., Man, Cybern. Syst.*, early access, Aug. 6, 2019, doi: 10.1109/TSMC.2019.2930525.
- [18] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
- [19] X. Luo, M. Zhou, S. Li, Z. You, Y. Xia, and Q. Zhu, "A nonnegative latent factor model for large-scale sparse matrices in recommender systems via alternating direction method," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 3, pp. 579–592, Mar. 2016.

- [20] B. Smith and G. Linden, "Two decades of recommender systems at Amazon.com," *IEEE Internet Comput.*, vol. 21, no. 3, pp. 12–18, May/Jun. 2017.
- [21] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proc. 39th Int.* ACM SIGIR Conf. Res. Develop. Inf. Retr., Jul. 2016, pp. 549–558.
- [22] W. Ma, N. Li, Y. Li, J. Duan, and B. Chen, "Sparse normalized least mean absolute deviation algorithm based on unbiasedness criterion for system identification with noisy input," *IEEE Access*, vol. 6, pp. 14379–14388, 2018.
- [23] Q. Ke and T. Kanade, "Robust subspace computation using L_1 norm," School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, 2003.
- [24] X. Zhu *et al.*, "Similarity-maintaining privacy preservation and locationaware low-rank matrix factorization for QoS prediction based Web service recommendation," *IEEE Trans. Services Comput.*, early access, May 23, 2019, doi: 10.1109/TSC.2018.2839741.
- [25] L. Wang, M. Gordon, and J. Zhu, "Regularized least absolute deviations regression and an efficient algorithm for parameter tuning," in *Proc. 6th Int. Conf. Data Mining (ICDM)*, Dec. 2006, pp. 690–700.
- [26] R. Koenker and K. Hallock, "Quantile regression," J. Econ. Perspect., vol. 15, no. 4, pp. 143–156, 2001.
- [27] B. Lakshminarayanan, G. Bouchard, and C. Archambeau, "Robust Bayesian matrix factorisation," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 425–433.
- [28] C. Wu, W. Qiu, Z. Zheng, X. Wang, and X. Yang, "QoS prediction of Web services based on two-phase K-means clustering," in *Proc. IEEE Int. Conf. Web Services*, Jun. 2015, pp. 161–168.
- [29] T. Goldstein and S. Osher, "The split Bregman method for L₁regularized problems," *SIAM J. Imag. Sci.*, vol. 2, no. 2, pp. 323–343, Jan. 2009.
- [30] X. Ren, M. Song, H. E, and J. Song, "Context-aware probabilistic matrix factorization modeling for point-of-interest recommendation," *Neurocomputing*, vol. 241, pp. 38–55, Jun. 2017.
- [31] K. Yu, S. Zhu, J. Lafferty, and Y. Gong, "Fast nonparametric matrix factorization for large-scale collaborative filtering," in *Proc. 32nd Int.* ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR), 2009, pp. 211–218.
- [32] H. Wu, Z. Zhang, K. Yue, B. Zhang, J. He, and L. Sun, "Dual-regularized matrix factorization with deep neural networks for recommender systems," *Knowl.-Based Syst.*, vol. 145, pp. 46–58, Apr. 2018.
- [33] C. Wang et al., "Confidence-aware matrix factorization for recommender systems," in Proc. 32nd AAAI Conf. Artif. Intell., 2018, pp. 434–442.
- [34] D. Wu, X. Luo, M. Shang, Y. He, G. Wang, and X. Wu, "A datacharacteristic-aware latent factor model for Web services QoS prediction," *IEEE Trans. Knowl. Data Eng.*, early access, Aug. 5, 2020, doi: 10.1109/TKDE.2020.3014302.
- [35] Y. Chen, H. Xu, C. Caramanis, and S. Sanghavi, "Robust matrix completion and corrupted columns," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, 2011, pp. 873–880.
- [36] Y. Cherapanamjeri, K. Gupta, and P. Jain, "Nearly optimal robust matrix completion," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 797–805.
- [37] O. Klopp, K. Lounici, and A. B. Tsybakov, "Robust matrix completion," *Probab. Theory Rel. Fields*, vol. 169, nos. 1–2, pp. 523–564, 2014.
- [38] H. Mansour, D. Tian, and A. Vetro, "Factorized robust matrix completion," in *Handbook of Robust Low-Rank and Sparse Matrix Decomposition: Applications in Image and Video Processing*, vol. 5. New York, NY, USA: Chapman & Hall, 2016.
- [39] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in Proc. Adv. Neural Inf. Process. Syst., 2007, pp. 41–48.
- [40] Y. Han, Y. Yang, and X. Zhou, "Co-regularized ensemble for feature selection," in *Proc. 23rd Int. Joint Conf. Artif. Intell. (IJCAI)*, 2013, pp. 1380–1386.
- [41] D. Wu, Q. He, X. Luo, M. Shang, Y. He, and G. Wang, "A Posteriorneighborhood-regularized latent factor model for highly accurate Web service QoS prediction," *IEEE Trans. Services Comput.*, early access, Dec. 24, 2019.
- [42] M. Shang, X. Luo, Z. Liu, J. Chen, Y. Yuan, and M. Zhou, "Randomized latent factor model for high-dimensional and sparse matrices from industrial applications," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 1, pp. 131–141, Jan. 2019.
- [43] C. Leng, H. Zhang, G. Cai, I. Cheng, and A. Basu, "Graph regularized lp smooth non-negative matrix factorization for data representation," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 2, pp. 584–595, Mar. 2019.
- [44] Q. Shi, H. Lu, and Y.-M. Cheung, "Rank-one matrix completion with automatic rank estimation via L₁-norm regularization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4744–4757, Oct. 2018.

- [45] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.
- [46] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. World Wide Web Conf.*, 2017, pp. 173–182.
- [47] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 3203–3209.
- [48] S. Zhang, Y. Tay, L. Yao, B. Wu, and A. Sun, "DeepRec: An opensource toolkit for deep learning based recommendation," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 6581–6583.
- [49] S. Zhang, L. Yao, A. Sun, S. Wang, G. Long, and M. Dong, "NeuRec: On nonlinear transformation for personalized ranking," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 3669–3675.
- [50] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," ACM Comput. Surv., vol. 52, no. 1, p. 5, 2019.
- [51] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec: Autoencoders meet collaborative filtering," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 111–112.
- [52] Q. Wang, B. Peng, X. Shi, T. Shang, and M. Shang, "DCCR: Deep collaborative conjunctive recommender for rating prediction," *IEEE Access*, vol. 7, pp. 60186–60198, 2019.
- [53] P. Li, Z. Wang, Z. Ren, L. Bing, and W. Lam, "Neural rating regression with abstractive tips generation for recommendation," in *Proc. 40th Int.* ACM SIGIR Conf. Res. Develop. Inf. Retr., Aug. 2017, pp. 345–354.
- [54] X. He and T.-S. Chua, "Neural factorization machines for sparse predictive analytics," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Aug. 2017, pp. 355–364.
- [55] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, "Attentional factorization machines: Learning the weight of feature interactions via attention networks," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 3119–3125.
- [56] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, Feb. 2017, pp. 425–434.
- [57] D. H. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in *Proc.* 10th ACM Conf. Recommender Syst., 2016, pp. 233–240.
- [58] Z.-H. Zhou and J. Feng, "Deep forest: Towards an alternative to deep neural networks," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 3553–3559.
- [59] S. Rendle, L. Zhang, and Y. Koren, "On the difficulty of evaluating baselines: A study on recommender systems," 2019, pp. 1–19, arXiv:1905.01395. [Online]. Available: https://arxiv.org/abs/1905.01395
- [60] S. Rendle, W. Krichene, L. Zhang, and J. Anderson, "Neural collaborative filtering vs. Matrix factorization revisited," in *Proc. 14th ACM Conf. Recommender Syst.*, Sep. 2020, pp. 240–248.
- [61] W. Hoeffding, "Probability inequalities for sums of bounded random variables," J. Amer. Stat. Assoc., vol. 58, no. 301, pp. 13–30, Mar. 1963.
- [62] N. CesaBianchi and G. Lugosi, *Prediction, Learning, and Games*. New York, NY, USA: Cambridge Univ. Press, 2006.
- [63] W. Rudin, Principles of Mathematical Analysis. New York, NY, USA: McGraw-Hill, 1964.
- [64] P. Massa and P. Avesani, "Trust-aware recommender systems," in Proc. ACM Conf. Recommender Syst. (RecSys), 2007, pp. 17–24.
- [65] L. Brozovsky and V. Petricek, "Recommender system for online dating service," 2007, arXiv:cs/0703042. [Online]. Available: https://arxiv.org/abs/cs/0703042
- [66] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Inf. Retr.*, vol. 4, no. 2, pp. 133–151, 2001.
- [67] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, "GroupLens: Applying collaborative filtering to Usenet news," *Commun. ACM*, vol. 40, no. 3, pp. 77–87, 1997.
- [68] Yahoo. R2-Yahoo! Music User Ratings of Songs With Artist, Album, and Genre Meta Information, V. 1.0 (1.4 Gbyte & 1.1 Gbyte). Accessed: Apr. 3, 2021. [Online]. Available: https://webscope.sandbox. yahoo.com/catalog.php?datatype=r
- [69] X. He, T. Chen, M.-Y. Kan, and X. Chen, "TriRank: Review-aware explainable recommendation by modeling aspects," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 1661–1670.
- [70] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," J. Mach. Learn. Res., vol. 7, pp. 1–30, Jan. 2006.
- [71] B. Rosner, R. J. Glynn, and M.-L.-T. Lee, "The wilcoxon signed rank test for paired comparisons of clustered data," *Biometrics*, vol. 62, no. 1, pp. 185–192, Mar. 2006.

- [72] D. Wu, X. Luo, G. Wang, M. Shang, Y. Yuan, and H. Yan, "A highly accurate framework for self-labeled semisupervised classification in industrial applications," *IEEE Trans. Ind. Informat.*, vol. 14, no. 3, pp. 909–920, Mar. 2018.
- [73] B. Recht, C. Re, S. Wright, and F. Niu, "Hogwild: A lock-free approach to parallelizing stochastic gradient descent," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 693–701.



Di Wu (Member, IEEE) received the Ph.D. degree in computer application technology from the Chongqing Institute of Green and Intelligent Technology (CIGIT), Chinese Academy of Sciences (CAS), Chongqing, China, in 2019.

He was a Visiting Scholar at the University of Louisiana, Lafayette, LA, USA, from April 2018 to April 2019. He is currently an Associate Professor with CIGIT, CAS. His research interests include machine learning and data mining. He has authored over 40 articles in the IEEE TRANSAC-

TIONS ON KNOWLEDGE DATA ENGINEERING, the IEEE TRANSACTIONS ON SYSTEMS, MAN, CYBERNETIC: SYSTEMS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE TRANSACTIONS ON SERVICES COM-PUTING, the IEEE International Conference on Data Mining, the ACM International World Wide Web Conferences, and the International Joint Conference on Artificial Intelligence.

Dr. Wu is currently serving as an Associate Editor for the *Frontiers in Neurorobotics*.



Mingsheng Shang received the Ph.D. degree in computer science from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2007.

He joined the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing, China, in 2015, and is currently a Professor of computer science and engineering. His research interests include data mining, complex networks, cloud computing, and their applications.



Xin Luo (Senior Member, IEEE) 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.

In 2016, he joined the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing, China, as a Professor of computer science. He is currently also a Distinguished Professor of computer science with the

Dongguan University of Technology, Dongguan, China. His current research interests include big data analysis and intelligent control. He has authored over 100 articles (including over 60 IEEE Transactions articles).

Dr. Luo was a recipient of the Hong Kong Scholar Program jointly by the Society of Hong Kong Scholars and China Post-Doctoral Science Foundation in 2014, the Pioneer Hundred Talents Program of Chinese Academy of Sciences in 2016, and the Advanced Support of the Pioneer Hundred Talents Program of Chinese Academy of Sciences in 2018. He is currently serving as an Associate Editor for the IEEE/CAA JOURNAL OF AUTOMATICA SINICA, the IEEE ACCESS, and the *Neurocomputing*. He has received the Outstanding Associate Editor reward from the IEEE/CAA Journal of Automatica Sinica in 2020 and from the IEEE ACCESS in 2018.



Zidong Wang (Fellow, IEEE) was born in Jiangsu, China, in 1966. He received the B.Sc. degree in mathematics from Suzhou University, Suzhou, China, in 1986, and the M.Sc. degree in applied mathematics and the Ph.D. degree in electrical engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1990 and 1994, respectively.

He is currently a Professor of dynamical systems and computing with the Department of Computer Science, Brunel University London, London, U.K.

His research interests include dynamical systems, signal processing, bioinformatics, control theory, and their applications. He has authored more than 600 articles in international journals. He is a holder of the Alexander von Humboldt Research Fellowship of Germany, the JSPS Research Fellowship of Japan, and the William Mong Visiting Research Fellowship of Hong Kong.

Dr. Wang is a member of the Academia Europaea and program committee for many international conferences and a fellow of the Royal Statistical Society. He serves (or has served) as the Editor-in-Chief for the *International Journal of Systems Science*, the Editor-in-Chief for the *Neurocomputing*, and an Associate Editor for 12 international journals including the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, the IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, and the IEEE TRANSACTIONS ON SIGNAL PROCESSING.