Efficient and Differentiable Low-rank Matrix Completion with Back Propagation

Zhaoliang Chen, Jie Yao, Guobao Xiao and Shiping Wang

***Abstract*—The low-rank matrix completion has gained rapidly increasing attention from researchers in recent years for its efficient recovery of the matrix in various fields. Numerous studies have exploited the popular neural networks to yield low-rank outputs under the framework of low-rank matrix factorization. However, due to the discontinuity and nonconvexity of rank function, it is difficult to directly optimize the rank function via back propagation. Although a large number of studies have attempted to find relaxations of the rank function, e.g., Schatten-p norm, they still face the following issues when updating parameters via back propagation: 1) These methods or surrogate functions are still non-differentiable, bringing obstacles to deriving the gradients of trainable variables. 2) Most of these surrogate functions perform singular value decomposition upon the original matrix at each iteration, which is time-consuming and blocks the propagation of gradients. To address these problems, in this paper, we develop an efficient block-wise model dubbed differentiable low-rank learning (DLRL) framework that adopts back propagation to optimize the Multi-Schatten-p norm Surrogate (MSS) function. Distinct from the original optimization of this surrogate function, the proposed framework avoids singular value decomposition to admit the gradient propagation and builds a block-wise learning scheme to minimize values of Schatten-p norms. Accordingly, it speeds up the computation and makes all parameters in the proposed framework learnable according to a predefined loss function. Finally, we conduct substantial experiments in terms of image recovery and collaborative filtering. The experimental results verify the superiority of the proposed framework in terms of both runtimes and learning performance compared with other state-of-the-art low-rank optimization methods. Our codes are available at https://github.com/chenzl23/DLRL.**

***Index Terms*—Low-rank matrix completion, back propagation, image recovery, collaborative filtering, Schatten-p norm.**

I. Introduction

LOW-RANK matrix optimization is a crucial technique in machine learning, which has been broadly studied by researchers in various fields, including recommender systems [1], [2], computer vision [3], [4] and low-rank representation [5], [6]. Low-rank learning has also been extensively explored in multimedia scenarios. For instance, Wang et al. extracted a shared low-rank correlation embedding to conduct multi-feature fusion in subspace learning fields [7]. Jing et al. studied low-rank multi-view embedding to facilitate the performance of micro-video popularity prediction [8]. A tripartite graph regularized latent low-rank representation approach was presented to predict fashion compatibility in shopping websites [9]. As an example, existing research and practice have validated that rating matrices in recommender systems and pixel matrices of RGB images are generally low-rank, because the primary information of these matrices is dominated by top singular values, which only account for a small part of all singular values. Therefore, missing values of incomplete matrices can be recovered by leveraging the low-rank property.

Generally low-rank matrix completion can be realized by matrix factorization approaches and rank optimization approaches, respectively. Matrix factorization approaches decompose the original incomplete matrix into two or multiple factors that are low-rank, and recover the missing entries in the original matrix with the product of these factors. A large amount of research has been devoted to this field [10], [11]. As to the rank optimization approaches, because rank minimization problems are nonconvex and discontinuous, they are generally NP-hard and difficult to cope with. Numerous surrogate approximation functions have been employed in addressing this problem. Theoretical analyses have revealed that the nuclear norm is a convex lower bound of the rank function, which sums up all singular values of the matrix [12], [13], [14]. In other words, the nuclear norm is an ℓ_1-norm relaxation on singular values compared with the rank operator that is relevant to ℓ_0-norm. The optimization problem raised by nuclear norm-based methods can be solved by the Lagrangian multiplier method [15], alternating direction method of multiplier [16] or proximal mapping algorithm [17]. However, these nuclear norm-based methods ordinarily lead to suboptimal performance because they may not be a suitable rank relaxation of ℓ_0-norm in some practical applications [18].

To overcome these drawbacks, a lot of nonconvex relaxation functions for rank minimization have been proposed to better approximate the rank function. Some nonconvex surrogates of ℓ_0-norm including ℓ_p-norm [19], Minimax Concave Plus (MCP) [20] and Smoothly Clipped Absolute Deviation (SCAD) [21] have been extended to relax the rank function, as listed in [22]. A number of researchers developed the nuclear norm with truncated nuclear norm-based or weighted nuclear norm-based methods [23], [24]. In addition, Schatten-p norm which unifies different norm formulations, has been widely investigated in rank optimization problems. The weighted

This work was supported in part by the National Natural Science Foundation of China under Grant U1705262 and the Natural Science Foundation of Fujian Province under Grants 2020J01130193 and 2018J07005. (Corresponding author: Shiping Wang).

Zhaoliang Chen, Jie Yao and Shiping Wang are with the College of Computer and Data Science, Fuzhou University, Fuzhou 350116, China and also with the Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350116, China (email: chenzl23@outlook.com, jieyao926@163.com, shipingwang-phd@163.com).

Guobao Xiao is with the College of Computer and Control Engineering, Minjiang University, Fuzhou 350108, China and also with the Fujian Provincial Key Laboratory of Information Processing and Intelligent Control, Minjiang University, Fuzhou 350108, China (email: gbx@mju.edu.cn).

Schatten-p norm minimization was put forward to tackle the applications to image denoising and background subtraction [25]. The multi-channel weighted Schatten-p norm minimization method was presented for image denoising upon RGB channels [26]. A modified Schatten-p norm in the affinity matrix rank minimization problem was developed to solve image recovery and recommender system problems [27]. It is noteworthy that the widely employed nuclear norm is a special case of Schatten-p norm when $p = 1$. Some prior works have pointed out that Schatten-p norm was a tighter approximation of rank function when $p \to 0$ [28]. Thus, Schatten-p norm is beneficial to constructing a more generalizing low-rank optimization framework. Experimental results of previous works also demonstrated that algorithms with Schatten-p norm outperformed nuclear norm-based methods [27], [29], [30].

However, it is universally acknowledged that most of these Schatten-p norm-based methods are confronted with a tricky problem that these optimization procedures entail conducting singular value decomposition (SVD) of the matrix at each iteration. This leads to severe time complexity and is inefficient for the matrix with high dimensions. Considerable neural network-based frameworks also require to yield low-rank output to conduct basic machine learning tasks like low-rank matrix completion [31]. In general, these methods were developed via the matrix factorization techniques, which decomposed the original matrix into two or multiple low-rank matrices [32]. Indeed, these frameworks only implicitly optimize the matrix rank under the assumption that data matrix should be low-rank, lacking a theoretical guarantee for minimizing the values of rank functions or other surrogates. However, back propagation that is widely used in neural network-based methods generally can not handle the rank optimization problem due to the nondifferentiability of SVD computation. To our knowledge, very limited research has been devoted to minimizing values of rank function with back propagation.

Some previous studies have attempted to construct neural networks or differentiable block-wise learning strategies according to some well-established optimization methods, which inspire us to convert the traditional iterative optimization algorithm into differentiable learning steps. Gregor and LeCun proposed Learned Iterative Shrinkage Thresholding Algorithm (LISTA) to unfold ISTA into block-wise neural networks and considered some variables as trainable weights without bias, which was applied to solve sparse coding problems [33]. Inspired by LISTA, Zhang et al. also cast ISTA into a deep learning form according to proximal mapping that optimized sparsity-inducing regularizer in a distinct way [34], termed as ISTA-Net. Yang et al. projected the iterations of ADMM into a data flow graph, according to which they established a deep ADMM-Net [35]. Du et al. constructed differentiable neural networks inspired by alternating iterative optimization and applied it to multi-view co-clustering [36]. Xie et al. reviewed linearized ADMM and replaced some intermediate variables with trainable layers to present the Differentiable Linearized ADMM (D-LADMM) [37]. Wang et al. transformed the proximal gradient method into differentiable neural networks to learn data-driven sparse regularizers [38]. These methods succeeded in obtaining remarkable performance of some well-known optimization problems via setting variables as trainable parameters and updating them via back propagation.

In this paper, we propose an efficient end-to-end learning framework to solve the matrix completion problem with optimization on Multi-Schatten-p norm Surrogate (MSS) function, which copes with the rank optimization problem explicitly over multiple factorized factors. In order to avoid conducting SVD at every iteration, we reformulate the proximal mapping problem with Lagrangian function and convert it into differentiable learning steps which contain some trainable variables. Furthermore, the optimization of MSS is reconstructed by the block-wise learning strategy, and all parameters are learned via back propagation. We also analyze the convergence of Schatten-p norm for the proposed method to ensure that DLRL can yield low-rank outputs. The main contribution of this paper includes the following three aspects:

1) Block-wise differentiable learning steps are proposed to handle the low-rank matrix recovery problem, dubbed Differentiable Low-Rank Learning (DLRL) framework.

2) Transform the universally utilized proximal mapping methods into a basic learning scheme without conducting SVD of inputs, which accelerates the computational speed and allows back propagation.

3) As applications to image recovery and collaborative filtering, substantial experimental results indicate the superior performance of the proposed framework.

The rest of this paper proceeds as follows. A brief summary of the relevant concepts and works in low-rank matrix completion are recalled in Section II. In Section III, a differentiable learning method for low-rank matrix completion is proposed based on MSS function, following theoretical analyses and discussion. Comprehensive experiments on synthetic data and practical applications are carried out to validate the effectiveness of the proposed DLRL in Section IV. Finally, we conclude our work in Section V.

II. Related Work

A. Low-Rank Matrix Factorization Methods

With the assumption that data matrices in various machine learning fields should be low-rank, a large number of matrix factorization-based methods have been developed. The Non-negative Matrix Factorization (NMF) algorithm [39] was conducted by minimizing the distance as

$$\arg\min_{\mathbf{U},\mathbf{V}\geq 0} \|\mathbf{X} - \mathbf{U}\mathbf{V}^T\|_F^2, \tag{1}$$

where $\mathbf{U} \in \mathbb{R}^{m\times d}$ and $\mathbf{V} \in \mathbb{R}^{n\times d}$ are non-negative matrices. Due to the reason that essential information is generally encoded in a low-rank intrinsic data matrix, the factorization of matrix demands for $d \ll \min(m, n)$.

In decades, a large number of methods for low-rank matrix factorization have been explored. Lu et al. improved NMF by considering structural incoherence and low-rank properties of image data, and applied it to image classification [40]. Wang et al. proposed the Diverse NMF (DiNMF) method to reduce the redundancy among multi-view representations [41]. Nguyen et al. put forward a graph neural network-based framework to exploit the underlying features by low-rank matrix factorization [2]. Wang et al. incorporated rank optimization into a generalized low-rank matrix factorization method [42]. Low-rank matrix factorization was leveraged to compress the word embedding layers for deep learning models of natural language processing [43]. Yang et al. introduced a tensor factorization technique into the multi-view deep autoencoder to address multi-view representation learning [32]. Arora et al. pointed out that increasing depth to deep matrix factorization enhanced an implicit tendency towards low-rank solutions [31]. In a nutshell, low-rank matrix factorization methods implicitly minimize the rank of data matrices by exploring decomposed low-dimensional matrices.

B. Rank Optimization Methods

Matrix factorization-based methods optimize the rank of matrix implicitly, while rank optimization methods directly minimize approximations of the rank function. A generalized rank optimization problem is formulated as

$$h(\mathbf{X}) = f(\mathbf{X}) + g(\mathbf{X}), \tag{2}$$

where $f(\mathbf{X})$ is a differentiable function and $g(\mathbf{X})$ is a nonconvex and non-smooth regularization enforcing low-rank constraint. A widely used rank regularization is the extended Schatten-p norm, defined by

$$\|\mathbf{X}\|_{S_p} = \left(\sum_{i=1}^{\min\{m,n\}} \sigma_i(\mathbf{X})^p\right)^{\frac{1}{p}} = \left(Tr((\mathbf{X}^T\mathbf{X})^{\frac{p}{2}})\right)^{\frac{1}{p}}, \tag{3}$$

where $0 < p < \infty$ and $\sigma_i(\mathbf{X})$ is the i-th singular value of matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$. Then the general Schatten-p norm minimization problem is denoted by

$$\arg\min_{\mathbf{X}} h(\mathbf{X}) = f(\mathbf{X}) + \|\mathbf{X}\|_{S_p}^p. \tag{4}$$

In light of these definitions, Schatten-p norm is an ℓ_p-norm constraint on singular values when $0 < p < 1$. In particular, it becomes widely employed nuclear norm or trace norm when $p = 1$ and rank norm when $p = 0$. This problem can be solved by proximal gradient descent algorithm [44], [37], where the proximal mapping of $\|\mathbf{X}\|_{S_p}^p$ is defined as

$$\mathbf{Prox}_{S_p}(\mathbf{Y}) = \arg\min_{\mathbf{X}} \frac{1}{2}\|\mathbf{X}-\mathbf{Y}\|_F^2 + \frac{1}{p}\|\mathbf{X}\|_{S_p}^p. \tag{5}$$

To reduce time consumption, some existing methods have attempted to search for surrogates of a specific p value with matrix factorization. For instance, the surrogate for nuclear norm ($p = 1$) [45], [46] has been widely investigated, defined by

$$\|\mathbf{X}\|_* = \arg\min_{\mathbf{U},\mathbf{V}:\mathbf{X}=\mathbf{U}\mathbf{V}^T} \frac{1}{2}\|\mathbf{U}\|_F^2 + \frac{1}{2}\|\mathbf{V}\|_F^2, \tag{6}$$

where $\mathbf{U} \in \mathbb{R}^{m\times d}$ and $\mathbf{V} \in \mathbb{R}^{n\times d}$ with $d \ll \min(m,n)$. Recent studies also proved the equality between decomposing-based surrogate and Schatten-p norm when $p = \frac{2}{3}$ and $\frac{1}{2}$ [47], [48], [49], as listed below:

$$\begin{aligned} \frac{3}{2}\|\mathbf{X}\|_{S_{2/3}}^{2/3} &= \min_{\mathbf{U},\mathbf{V}:\mathbf{X}=\mathbf{U}\mathbf{V}^T} \|\mathbf{U}\|_* + \frac{1}{2}\|\mathbf{V}\|_F^2, \\ 2\|\mathbf{X}\|_{S_{1/2}}^{1/2} &= \min_{\mathbf{U},\mathbf{V}:\mathbf{X}=\mathbf{U}\mathbf{V}^T} \|\mathbf{U}\|_* + \|\mathbf{V}\|_*. \end{aligned} \tag{7}$$

Based on these previous studies, multi-Schatten-p norm surrogate was investigated to unify these different decomposing-based surrogates [28]. This inspires us to establish a block-wise neural network where each block addresses a parameterized factor as a subproblem.

III. Differentiable Low-Rank Learning

In this section, we provide the details of the proposed DLRL framework. For better understanding, primarily used mathematical notations are recalled in advance. In the field of low-rank matrix completion, $\mathbf{X} \in \mathbb{R}^{m\times n}$ is the incomplete matrix, and the rank of matrix is defined by $\mathrm{rank}(\cdot)$. The observed matrix is denoted by $\mathbf{X} \odot \mathbf{H}$, where $\mathbf{H}$ is the binary indicator matrix and $\odot$ represents the Hadamard product. In this paper, we approximate the rank by adopting Schatten-p norm $\|\mathbf{X}\|_{S_p}$ with the indicator p. With matrix factorization, the low-rank approximated matrix $\hat{\mathbf{X}}$ is factorized into I factor matrices $\{\hat{\mathbf{X}}_i\}_{i=1}^I$.

A. Multi-Schatten-p Norm Surrogate

Firstly, we review the MSS function [28]. Assuming that $\mathbf{X}_i, i = 1, \cdots, I$ are matrices where $\mathbf{X}_1 \in \mathbb{R}^{m\times d_1}$, $\mathbf{X}_i \in \mathbb{R}^{d_{i-1}\times d_i}, i = 2, \cdots, I-1$, $\mathbf{X}_I \in \mathbb{R}^{d_{I-1}\times n}$, and denoting $\mathbf{X} = \prod_{i=1}^I \mathbf{X}_i$, $\mathrm{rank}(\mathbf{X}) = r \le \min\{d_i, i = 1, \cdots, I\}$, we have

$$\frac{1}{p}\|\mathbf{X}\|_{S_p}^p = \min_{\mathbf{X}=\prod_{i=1}^I \mathbf{X}_i} \sum_{i=1}^I \frac{1}{p_i}\|\mathbf{X}_i\|_{S_{p_i}}^{p_i}, \tag{8}$$

where any $p_i > 0$ satisfies $\frac{1}{p} = \sum_{i=1}^I \frac{1}{p_i}$. Considering Equation (8) as a surrogate of Schatten-p norm, minimization problem defined in Equation (4) is transformed into

$$\arg\min_{\mathbf{X}=\prod_{i=1}^I \mathbf{X}_i} h(\mathbf{X}) = f(\mathbf{X}) + \sum_{i=1}^I \frac{1}{p_i}\|\mathbf{X}_i\|_{S_{p_i}}^{p_i}. \tag{9}$$

This optimization problem can be solved by Block Coordinate Descent (BCD) [50] which minimizes each $\mathbf{X}_i$ of Equation (9) at a single iteration by fixing the remaining blocks. At each iteration, the proximal gradient method for each factor $\mathbf{X}_i$ is as follows:

$$\begin{aligned} \mathbf{X}_i^{(k+1)} &= \arg\min_{\mathbf{X}_i} f(\mathbf{X}_i^{(k)}) + \langle \nabla f(\mathbf{X}_i^{(k)}), \mathbf{X}_i - \mathbf{X}^{(k)}\rangle \\ &\quad + \frac{L_i^{(k-1)}}{2}\left\|\mathbf{X}_i - \mathbf{X}_i^{(k)}\right\|_F^2 + \frac{1}{p_i}\|\mathbf{X}_i\|_{S_{p_i}}^{p_i} \\ &= \arg\min_{\mathbf{X}_i} \frac{L_i^{(k-1)}}{2}\|\mathbf{X}_i - \mathbf{Y}_i\|_F^2 + \frac{1}{p_i}\|\mathbf{X}_i\|_{S_{p_i}}^{p_i}, \end{aligned} \tag{10}$$

where $\mathbf{Y}_i = \mathbf{X}_i^{(k)} - \frac{1}{L_i^{(k-1)}}\nabla f(\mathbf{X}_i^{(k)})$ and $L_i^{(k-1)}$ is the Lipschitz constant of $\nabla f(\cdot)$, satisfying

$$\|\nabla f(\mathbf{X}) - \nabla f(\mathbf{Y})\|_F \le L_i^{(k-1)}\|\mathbf{X}-\mathbf{Y}\|_F, \tag{11}$$

for any $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{n\times m}$. The optimization problem defined in Equation (10) can be exactly solved by the proximal operation that is related to Schatten-p norm, as follows:

$$\mathbf{X}_i^{(k+1)} = \mathbf{Prox}_{S_{p_i}}\left(\mathbf{X}_i^{(k)} - \frac{1}{L_i^{(k-1)}}\nabla f(\mathbf{X}_i^{(k)})\right). \tag{12}$$

Because Schatten-p norm is an ℓ_p constraint on singular values when $0 < p < 1$, Equation (12) can be solved by some ℓ_p-norm shrinkage algorithms or closed-form solutions for some specific p_i values. However, it is time-consuming because of SVD at each iteration. As a matter of fact, a host of similar methods based on Schatten-p norm function also suffer from this issue.

B. Optimizing Multi-Schatten-p Norm Surrogate Function with Neural Networks

We notice that the BCD method to optimize MSS function is a block-wise solver naturally. It is similar to the block-wise neural network structure in deep learning where each block is differentiable and reusable. This motivates us to build a similar end-to-end structure for learning low-rank constraints with back propagation and gradient descent. In the first place, the optimization of the matrix completion problem with MSS function is formulated as

$$\arg\min_{\mathbf{X}_i, i=1,\cdots,I} \frac{1}{2}\left\| P_\Omega\left(\prod_{i=1}^{I}\mathbf{X}_i\right) - P_\Omega(\mathbf{X})\right\|_F^2 + \sum_{i=1}^{I}\frac{1}{p_i}\|\mathbf{X}_i\|_{S_{p_i}}^{p_i}, \tag{13}$$

where $\mathbf{X} \in \mathbb{R}^{m\times n}$ is a given observed matrix. The projection $P_\Omega(\mathbf{X}) = \mathbf{H}\odot\mathbf{X}$ indicates the entries for training. With $f(\mathbf{X}_1,\cdots,\mathbf{X}_I) = \frac{1}{2}\left\|P_\Omega\left(\prod_{i=1}^{I}\mathbf{X}_i\right) - P_\Omega(\mathbf{X})\right\|_F^2$ and $g(\mathbf{X}_1,\cdots,\mathbf{X}_I) = \sum_{i=1}^{I}\frac{1}{p_i}\|\mathbf{X}_i\|_{S_{p_i}}^{p_i}$, the gradient of $f(\mathbf{X}_i)$ at $\mathbf{X}_i$ is given by

$$\nabla f(\mathbf{X}_i) = (\mathbf{X}_1\cdots\mathbf{X}_{i-1})^T\left(\mathbf{H}\odot\left(\mathbf{X}-\prod_{i=1}^{I}\mathbf{X}_i\right)\right)(\mathbf{X}_{i+1}\cdots\mathbf{X}_I)^T. \tag{14}$$

Inspired by BCD, the proposed method is constructed as a framework with I differentiable neural network blocks, where every block copes with a similar subproblem by going through the same computation progress depending on Equations (12) and (14).

Because the global Schatten-p norm minimization is transformed into multiple subproblems solved with blocks, we now only consider the local Schatten-p norm optimization problem for each independent $\mathbf{X}_i$ in one block. Since it is problematic to cope with Equation (12) with neural networks, whose solutions are generally based on nondifferentiable SVD, we have to explore another way to solve Equation (10). Optimization of Equation (10) can be viewed as finding approximate $\mathbf{X}_i$ for $\mathbf{Y}_i$ with low-rank constraint. Therefore, we rewrite Equation (10) with a constraint, as shown below:

$$\arg\min_{\mathbf{Y}_i}\frac{1}{p_i}\|\mathbf{Y}_i\|_{S_{p_i}}^{p_i}, \qquad \text{s.t.} \quad \mathbf{Y}_i = \mathbf{X}_i, \tag{15}$$

where we directly optimize $\mathbf{Y}_i$. Actually, Equation (15) is a strict version of optimization problem defined in Equation (10). Associated with Equation (3), the solution of Equation (15) can be derived from the Lagrangian function, which is defined by

$$\mathcal{L}(\mathbf{Y}_i,\mathbf{\Lambda}_i) = \frac{1}{p_i}Tr(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i}{2}} - Tr(\mathbf{\Lambda}_i^T(\mathbf{Y}_i-\mathbf{X}_i)), \tag{16}$$

where $\mathbf{\Lambda}_i$ is the Lagrangian multiplier. Computing the derivative of $\mathcal{L}(\mathbf{Y}_i,\mathbf{\Lambda}_i)$ w.r.t. $\mathbf{Y}_i$ and setting the derivative to zero, we have

$$\frac{\partial\mathcal{L}(\mathbf{Y}_i,\mathbf{\Lambda}_i)}{\partial\mathbf{Y}_i} = 2\mathbf{Y}_i\cdot\frac{1}{2}(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}} - \mathbf{\Lambda}_i = 0, \tag{17}$$

that is,

$$\mathbf{\Lambda}_i = 2\mathbf{Y}_i\cdot\frac{1}{2}(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}}. \tag{18}$$

However, it is challenging to obtain a solution of Equation (18) w.r.t. $\mathbf{Y}_i$. Therefore, we consider $\frac{1}{2}(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}}$ as a constant. Letting $\mathbf{G}_i = \frac{1}{2}(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}}$, we obtain

$$\mathbf{\Lambda}_i = 2\mathbf{Y}_i\mathbf{G}_i, \tag{19}$$

from which we can directly achieve an approximation of $\mathbf{Y}_i$ via $\mathbf{\Lambda}_i$ and $\mathbf{G}_i$. According to Equation (19) and the constraint $\mathbf{Y}_i = \mathbf{X}_i$, we have

$$\mathbf{\Lambda}_i = 2\mathbf{X}_i\mathbf{G}_i, \tag{20}$$

which constructs the correspondence between the Lagrangian multiplier $\mathbf{\Lambda}_i$ and approximate $\mathbf{X}_i$. Namely,

$$\mathbf{X}_i = \frac{1}{2}\mathbf{\Lambda}_i\mathbf{G}_i^{-1}, \tag{21}$$

where $\mathbf{X}_i$ is calculated via the current $\mathbf{G}_i$.

It is widely acknowledged that neural network-based framework is one of the most promising technologies, which has significantly improved the performance of learning tasks compared with traditional optimization schemes. Intuitively, this inspires us to directly learn $\mathbf{X}_i$ with learnable $\mathbf{\Lambda}_i$ according to Equation (21), which maps the updating rules derived from Lagrangian function to a neural network structure. Differentiability with regard to Lagrangian multiplier methods allows us to construct block-wise neural networks, so that derivatives can be obtained by applying a classic back propagation. Following spirits of learning-based optimization that we have deduced before, we develop an efficient learning framework for achieving estimated $\hat{\mathbf{X}}_i$, which reaches low-rank solutions and minimizes the reconstruction errors simultaneously. Some works have also considered replacing the variables in optimization steps with trainable layers to construct a differentiable neural network, achieving encouraging performance [33], [35], [37]. Considering that $\mathbf{\Lambda}_i$ is a learnable linear layer $\mathbf{W}_i$ without bias in neural networks, we have

$$\hat{\mathbf{X}}_i = \frac{1}{2}\mathbf{W}_i^{(k)}\mathbf{G}_i^{-1}, \tag{22}$$

where $\mathbf{W}_1^{(k)} \in \mathbb{R}^{m\times d_1}$, $\mathbf{W}_i^{(k)} \in \mathbb{R}^{d_{i-1}\times d_i}, i = 2,\cdots,I-1$ and $\mathbf{W}_I^{(k)} \in \mathbb{R}^{d_{I-1}\times n}$ are weights updated by back propagation at the k-th iteration. Here $\mathbf{G}_i^{-1}$ is the pseudo-inverse

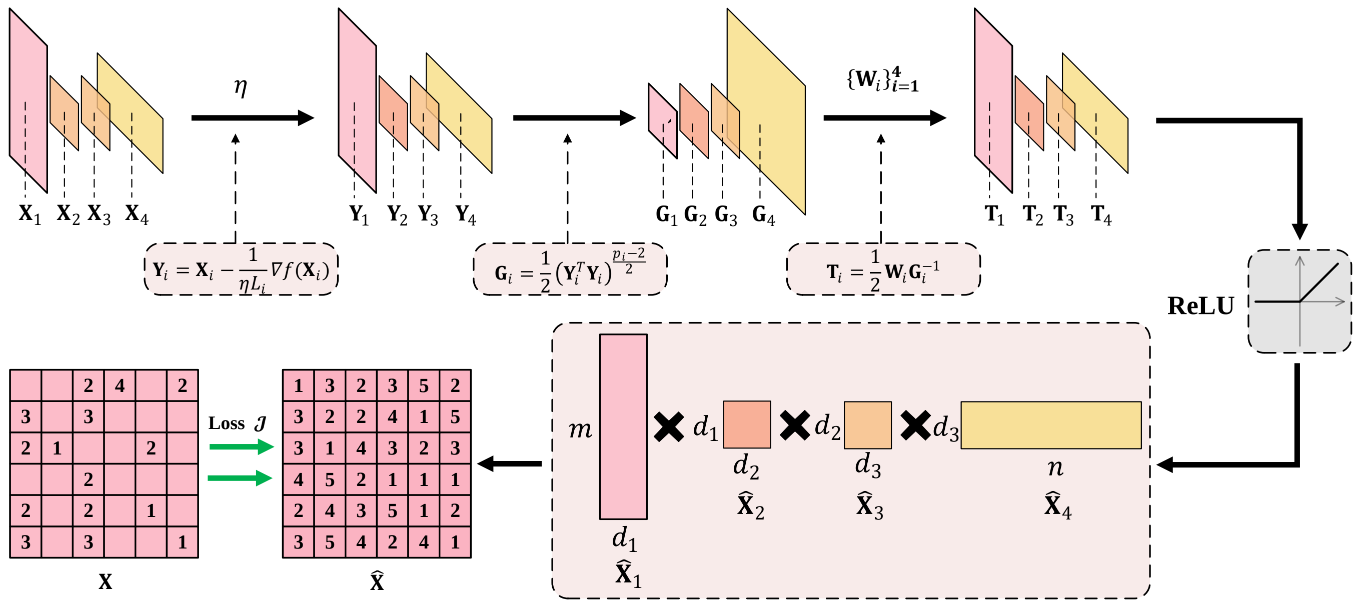
η
$\{\mathbf{W}_i\}_{i=1}^4$
$\mathbf{X}_1$ $\mathbf{X}_2$ $\mathbf{X}_3$ $\mathbf{X}_4$
$\mathbf{Y}_1$ $\mathbf{Y}_2$ $\mathbf{Y}_3$ $\mathbf{Y}_4$
$\mathbf{G}_1$ $\mathbf{G}_2$ $\mathbf{G}_3$ $\mathbf{G}_4$
$\mathbf{T}_1$ $\mathbf{T}_2$ $\mathbf{T}_3$ $\mathbf{T}_4$
$\mathbf{Y}_i = \mathbf{X}_i - \frac{1}{\eta L_i}\nabla f(\mathbf{X}_i)$
$\mathbf{G}_i = \frac{1}{2}(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}}$
$\mathbf{T}_i = \frac{1}{2}\mathbf{W}_i\mathbf{G}_i^{-1}$
ReLU
Loss $\mathcal{J}$
m d_1 d_2 d_3 n
$\hat{\mathbf{X}}_1$ $\hat{\mathbf{X}}_2$ $\hat{\mathbf{X}}_3$ $\hat{\mathbf{X}}_4$
$\mathbf{X}$
$\hat{\mathbf{X}}$

Fig. 1: Structure of the proposed DLRL with 4 blocks, which is a block-wise differentiable neural network framework. The illustration contains 5 learnable parameters, including a parameter η shared across all blocks and 4 weight matrices $\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3, \mathbf{W}_4$ in each block. The final estimated matrix $\hat{\mathbf{X}}$ is obtained by the multiplications of matrices learned in each block, that is, $\hat{\mathbf{X}} = \hat{\mathbf{X}}_1\hat{\mathbf{X}}_2\hat{\mathbf{X}}_3\hat{\mathbf{X}}_4$.

of matrix $\mathbf{G}_i$. Therefore, the neural network-based learning procedure for minimizing Equation (15) is

$$\begin{aligned}\mathbf{G}_i &= \frac{1}{2}(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}},\\ \mathbf{T}_i &= \frac{1}{2}\mathbf{W}_i^{(k)}\mathbf{G}_i^{-1},\\ \hat{\mathbf{X}}_i &= \mathrm{ReLU}(\mathbf{T}_i).\end{aligned} \tag{23}$$

Because most matrix factorization-based methods require the latent submatrices to be non-negative, we adopt the rectified linear unit function $\mathrm{ReLU}(\cdot)$ to ensure the non-negativity of outputs and maintain features of yielded submatrices. Computing Equation (23) can be viewed as a three-step differentiable low-rank learning operator that learns low-rank outputs in each block. Because the presented forward propagation rules are derived from Equation (20), they can be regarded as a solution of the linear equation, where $\mathbf{\Lambda}_i$ is approximately obtained via trainable $\mathbf{W}_i$.

To accelerate the convergence speed of networks, we add a new learnable parameter $\eta \in (0, 1)$ to control the updating rate of $\mathbf{Y}_i$ automatically, which is learned via back propagation during network training. It was inspired by the techniques of accelerated proximal gradient algorithms [51]. In order to guarantee that η satisfies $0 < \eta < 1$, we project it onto the corresponding domain before forward calculation at each epoch. Therefore, $\mathbf{Y}_i$ at each block is computed by

$$\mathbf{Y}_i = \mathbf{X}_i - \frac{1}{\eta L_i}\nabla f(\mathbf{X}_i). \tag{24}$$

In general, the parameter setup for the proposed method is $\mathbf{\Theta} = \{\eta, \mathbf{W}_1, \cdots, \mathbf{W}_I\}$. The Lipschitz constant is computed at each block by [52], [28]

$$L_i = \max\{\|\mathbf{X}_1\cdots\mathbf{X}_{i-1}\|_2^2\|\mathbf{X}_{i+1}\cdots\mathbf{X}_I\|_2^2, \epsilon\}. \tag{25}$$

The loss function of the proposed networks is calculated by

$$\mathcal{J}(\mathbf{X}, \hat{\mathbf{X}}_1, \cdots, \hat{\mathbf{X}}_I) = \frac{1}{2}\left\| P_\Omega\left(\prod_{i=1}^{I}\hat{\mathbf{X}}_i\right) - P_\Omega(\mathbf{X})\right\|_F^2, \tag{26}$$

which measures the reconstruction errors. Therefore, DLRL achieves the low-rank output with differentiable operator in each block and minimizes the reconstruction errors simultaneously. Algorithm 1 illustrates the training process of the proposed DLRL, where all learnable parameters are updated according to their gradients. The training process stops when the loss function $\mathcal{J}(\mathbf{X}, \hat{\mathbf{X}}_1, \cdots, \hat{\mathbf{X}}_I)$ converges. Figure 1 illustrates the structure of DLRL with 4 blocks as an example.

C. Back Propagation for DLRL

The gradients of learnable parameters using in back propagation are derived from the loss function (26). For simplicity, we denote the loss $\mathcal{J}(\mathbf{X}, \hat{\mathbf{X}}_1, \cdots, \hat{\mathbf{X}}_I)$ as $\mathcal{J}$ in this subsection. Because gradient computations are the same in each block, we only derive gradients in a single block as an example. Accordingly, we write $\mathbf{W}_i$ as $\mathbf{W}$, $\mathbf{T}_i$ as $\mathbf{T}$, L_i as L, $\mathbf{X}_i$ as $\mathbf{X}$, and $\hat{\mathbf{X}}_i$ as $\hat{\mathbf{X}}$, respectively. With the aforementioned notations, we derive the gradients of learnable parameters $\mathbf{W}$ and η in the i-th block as follows:

1) Gradient of $\mathbf{W}$*:* The coordinate-wise partial derivative of the loss function $\mathcal{J}$ w.r.t. $\mathbf{W}$ is expressed as

$$\frac{\partial\mathcal{J}}{\partial\mathbf{W}_{cr}} = \sum_{p,l}\frac{\partial\mathcal{J}}{\partial\mathbf{T}_{pl}}\frac{\partial\mathbf{T}_{pl}}{\partial\mathbf{W}_{cr}}, \tag{27}$$

where $\mathbf{W}_{cr}$ is the (c, r)-th entry of $\mathbf{W}$. Expanding the partial derivative $\frac{\partial\mathbf{T}_{pl}}{\partial\mathbf{W}_{cr}}$ leads to

$$\frac{\partial\mathbf{T}_{pl}}{\partial\mathbf{W}_{cr}} = \frac{1}{2}\frac{\partial\sum_s\mathbf{W}_{ps}\left(\mathbf{G}^{-1}\right)_{sl}}{\partial\mathbf{W}_{cr}} = \frac{1}{2}\frac{\partial\mathbf{W}_{pr}\left(\mathbf{G}^{-1}\right)_{rl}}{\partial\mathbf{W}_{cr}}, \tag{28}$$

Algorithm 1 Differentiable Low-Rank Learning (DLRL)

```
Input: Data matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$ and indicators of Schatten-$p$ norm $\{p_i\}_{i=1}^{I}$.
Output: The predicted factors $\hat{\mathbf{X}}_i, i = 1, \dots, I$.
1: Initialize learnable weight matrices $\mathbf{W}_i, i = 1, \dots, I$ and learnable parameter $\eta$;
2: Initialize factors $\mathbf{X}_i, i = 1, \dots, I$ and counter $k = 0$;
3: while $\mathcal{J}(\mathbf{X}, \hat{\mathbf{X}}_1, \cdots, \hat{\mathbf{X}}_I)$ does not converge or the framework does not meet the early-stop condition do
4:   for $i = 1 \to I$ do
5:     Obtain Lipschitz constant $L_i$ with Equation (25);
6:     Calculate input of the differentiable low-rank operator $\mathbf{Y}_i$ with Equations (14) and (24);
7:     Compute output of the differentiable low-rank operator with Equation (23);
8:   end for
9:   Learn trainable weights $\mathbf{W}_i^{(k)}, i = 1, \dots, I$ and updating rate $\eta^{(k)}$ with back propagation;
10:  Update counter $k = k + 1$;
11: end while
12: return $\hat{\mathbf{X}}_i, i = 1, \dots, I$.
```

which implies that

$$\frac{\partial \mathcal{J}}{\partial \mathbf{W}_{cr}} = \frac{1}{2}\sum_{p,l} \frac{\partial \mathcal{J}}{\partial \mathbf{T}_{pl}} \frac{\partial \mathbf{W}_{pr}\left(\mathbf{G}^{-1}\right)_{rl}}{\partial \mathbf{W}_{cr}}, \tag{29}$$

where gradient $\frac{\partial \mathbf{W}_{pr}}{\partial \mathbf{W}_{cr}} = 1$ only when $p = c$, and 0 otherwise. Therefore, this indicates that

$$\begin{aligned}\frac{\partial \mathcal{J}}{\partial \mathbf{W}_{cr}} &= \frac{1}{2}\sum_{l} \frac{\partial \mathcal{J}}{\partial \mathbf{T}_{cl}} \frac{\partial \mathbf{W}_{cr}\left(\mathbf{G}^{-1}\right)_{rl}}{\partial \mathbf{W}_{cr}} \\ &= \frac{1}{2}\sum_{l} \frac{\partial \mathcal{J}}{\partial \mathbf{T}_{cl}} \left(\mathbf{G}^{-1}\right)_{rl}.\end{aligned} \tag{30}$$

Transforming to the form of matrix, we have

$$\frac{\partial \mathcal{J}}{\partial \mathbf{W}} = \frac{1}{2}\frac{\partial \mathcal{J}}{\partial \mathbf{T}} \left(\mathbf{G}^{-1}\right)^{T}, \tag{31}$$

with

$$\frac{\partial \mathcal{J}}{\partial \mathbf{T}} = \frac{\partial \mathcal{J}}{\partial \hat{\mathbf{X}}} \nabla \mathrm{ReLU}(\mathbf{T}), \tag{32}$$

where we can compute the derivative $\frac{\partial \mathcal{J}}{\partial \hat{\mathbf{X}}} = \nabla f(\hat{\mathbf{X}})$ with Equation (14).

2) Gradient of η: As to the partial derivative of $\mathcal{J}$ w.r.t. η, it is computed by

$$\frac{\partial \mathcal{J}}{\partial \eta} = \mathrm{Tr}\left[\left(\frac{\partial \mathcal{J}}{\partial \mathbf{Y}}\right)^{T} \frac{\partial \mathbf{Y}}{\partial \eta}\right], \tag{33}$$

where

$$\frac{\partial \mathbf{Y}}{\partial \eta} = \frac{1}{L\eta^2}\nabla f(\mathbf{X}). \tag{34}$$

Further, we compute $\frac{\partial \mathcal{J}}{\partial \mathbf{Y}}$ by

$$\frac{\partial \mathcal{J}}{\partial \mathbf{Y}} = \frac{p-2}{2}\frac{\partial \mathcal{J}}{\partial \mathbf{G}} \mathbf{Y}\left(\mathbf{Y}^T\mathbf{Y}\right)^{\frac{p-4}{2}}. \tag{35}$$

Given $\mathcal{G} = \mathbf{G}^{-1}$, we compute the coordinate-wise $\frac{\partial \mathcal{J}}{\partial \mathcal{G}_{cr}}$ by

$$\frac{\partial \mathcal{J}}{\partial \mathcal{G}_{cr}} = \sum_{p,l} \frac{\partial \mathcal{J}}{\partial \mathbf{T}_{pl}} \frac{\partial \mathbf{T}_{pl}}{\partial \mathcal{G}_{cr}}, \tag{36}$$

where

$$\frac{\partial \mathbf{T}_{pl}}{\partial \mathcal{G}_{cr}} = \frac{1}{2}\frac{\partial \sum_s \mathbf{W}_{ps}\mathcal{G}_{sl}}{\partial \mathcal{G}_{cr}} = \frac{1}{2}\frac{\partial \mathbf{W}_{pc}\mathcal{G}_{cl}}{\partial \mathcal{G}_{cr}}. \tag{37}$$

Therefore, considering both Equations (36) and (37), we have

$$\frac{\partial \mathcal{J}}{\partial \mathcal{G}_{cr}} = \frac{1}{2}\sum_{p,l} \frac{\partial \mathcal{J}}{\partial \mathbf{T}_{pl}} \frac{\partial \mathbf{W}_{pc}\mathcal{G}_{cl}}{\partial \mathcal{G}_{cr}}. \tag{38}$$

Analogously, gradient vanishes when $l \neq r$, leading to

$$\frac{\partial \mathcal{J}}{\partial \mathcal{G}_{cr}} = \frac{1}{2}\sum_{p} \frac{\partial \mathcal{J}}{\partial \mathbf{T}_{pr}} \frac{\partial \mathbf{W}_{pc}\mathcal{G}_{cr}}{\partial \mathcal{G}_{cr}} = \frac{1}{2}\sum_{p} \frac{\partial \mathcal{J}}{\partial \mathbf{T}_{pr}} \mathbf{W}_{pc}. \tag{39}$$

Hence, the matrix form of derivative is

$$\frac{\partial \mathcal{J}}{\partial \mathcal{G}} = \frac{1}{2}\mathbf{W}^T \frac{\partial \mathcal{J}}{\partial \mathbf{T}}. \tag{40}$$

Accordingly, we have

$$\frac{\partial \mathcal{J}}{\partial \mathbf{G}} = -\frac{1}{2}\mathbf{W}^T \frac{\partial \mathcal{J}}{\partial \mathbf{T}} (\mathbf{G}^{-1})^T \mathbf{G}^{-1}. \tag{41}$$

D. Convergence Analysis

The framework of DLRL is comprised of multiple blocks, where each block deals with the same subproblem. Because the network structure is developed from the proximal gradient method and Lagrangian multiplier method, each block is differentiable with a theoretical guarantee. Thus, the convergence of DLRL is guaranteed by back propagation of neural networks. It is notable that the convergence of loss values can not guarantee the convergence of Schatten-p norm values. Because each block of DLRL is derived from optimization on subproblems correlated with Schatten-p norm, the block-wise network can minimize the value of Schatten-p norm by forward propagation during training iterations. Equation (26) is applied to make the framework output a matrix that approximates the incomplete matrix at the best, while the forward propagation minimizes the rank of each yielded submatrix. Therefore, we need to prove that the convergence of Schatten-p norm can be guaranteed by the forward calculation of neural networks. To prove the convergence of Schatten-p norm values in each block, we first introduce a lemma.

Lemma 1: [29] For any positive definite matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{d\times d}$, the following inequality holds when $0 < p_i \leq 2$:

$$\begin{aligned} Tr\left(\mathbf{A}^{\frac{p_i}{2}}\right) &- \frac{p_i}{2}Tr\left(\mathbf{A}\mathbf{B}^{\frac{p_i-2}{2}}\right) \\ &\leq Tr\left(\mathbf{B}^{\frac{p_i}{2}}\right) - \frac{p_i}{2}Tr\left(\mathbf{B}\mathbf{B}^{\frac{p_i-2}{2}}\right).\end{aligned} \tag{42}$$

Then we prove the following theorem.

Theorem 1: For any output $\hat{\mathbf{X}}_i, i = 1, \cdots, I$ of the i-th block in Algorithm 1, $\|\hat{\mathbf{X}}_i\|_{S_{p_i}}^{p_i} \leq \|\mathbf{Y}_i\|_{S_{p_i}}^{p_i}$ at each iteration.

Proof 1: Because $\hat{\mathbf{X}}_i = \frac{1}{2}\mathbf{W}_i^{(k)}\mathbf{G}_i^{-1}$ is the solution to following optimization problem:

$$\arg\min_{\mathbf{Y}_i} Tr(\mathbf{Y}_i^T\mathbf{Y}_i\mathbf{G}), \qquad \text{s.t.} \quad \mathbf{Y}_i = \mathbf{X}_i, \tag{43}$$

then we have

$$Tr(\hat{\mathbf{X}}_i^T\hat{\mathbf{X}}_i\mathbf{G}) \leq Tr(\mathbf{Y}_i^T\mathbf{Y}_i\mathbf{G}) \tag{44}$$

in the i-th block. Since $\mathbf{G}_i = \frac{1}{2}(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}}$, this indicates the inequality that

$$\frac{1}{2}Tr\left(\hat{\mathbf{X}}_i^T\hat{\mathbf{X}}_i(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}}\right) \leq \frac{1}{2}Tr\left(\mathbf{Y}_i^T\mathbf{Y}_i(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}}\right). \tag{45}$$

That is to say

$$\frac{p_i}{2}Tr\left(\hat{\mathbf{X}}_i^T\hat{\mathbf{X}}_i(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}}\right) \leq \frac{p_i}{2}Tr\left(\mathbf{Y}_i^T\mathbf{Y}_i(\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i-2}{2}}\right) \tag{46}$$

when $0 < p_i \leq 2$. Given $\mathbf{A} = \hat{\mathbf{X}}_i^T\hat{\mathbf{X}}_i$ and $\mathbf{B} = \mathbf{Y}_i^T\mathbf{Y}$, and considering both Equations (42) and (46), we have

$$Tr\left((\hat{\mathbf{X}}_i^T\hat{\mathbf{X}}_i)^{\frac{p_i}{2}}\right) \leq Tr\left((\mathbf{Y}_i^T\mathbf{Y}_i)^{\frac{p_i}{2}}\right). \tag{47}$$

Thus $\|\hat{\mathbf{X}}_i\|_{S_{p_i}}^{p_i} \leq \|\mathbf{Y}_i\|_{S_{p_i}}^{p_i}$, completing the proof when $0 < p_i \leq 2$.

The proof above provides the convergence guarantee of local Schatten-p norm value in each block. Because each block yields factorized matrix with lower Schatten-p norm value, according to Equation (8), the global values of Schatten-p norm $\frac{1}{p}\|\mathbf{X}\|_{S_p}^p$ can be minimized.

E. Discussions on DLRL

DLRL aims to solve the low-rank optimization problem by minimizing the MSS function, using differentiable block-wise neural networks. With previous analyses, DLRL can optimize Schatten-p norm-based problems with different values of p, as long as the block number I and indicators of Schatten-p norm $\{p_i\}_{i=1}^I$ in each block satisfy the following conditions:

1) $\frac{1}{p} = \sum_{i=1}^I \frac{1}{p_i}$;
2) $0 < p_i \leq 2$ for $i = 1, \cdots, I$.

For instance, if we want to approximate $p = \frac{1}{2}$, it is applicable to fix $\{p_i = 1\}_{i=1}^I$ with block number $I = 2$, or $\{p_i = \frac{3}{2}\}_{i=1}^I$ with block number $I = 3$. From Algorithm 1, we can observe that the main time consumption of DLRL is based on matrix multiplication and matrix inversion. To simplify, the dimensions of factors are all denoted as d. Therefore, the time complexity for forward propagation of all blocks is $\mathcal{O}((I-2)d^3 + md^2 + nd^2)$. Given that $d \ll \min(m, n)$, the overall time complexity is $\mathcal{O}((m+n)d^2)$. Because the dimension d of factors is much smaller than the dimension of original matrix, the DLRL framework is efficient. Compared with other rank minimization methods, DLRL avoids SVD ($\mathcal{O}(\min(m,n)mn)$) at every iteration so that further speeds up the framework, and allows back propagation of the neural networks.

Finally, we claim that the main differences between DLRL and the previous optimization on MSS [28] are that:

1) DLRL updates weight matrix $\mathbf{W}_i$ for each block instead of updating $\mathbf{X}_i$ directly in MSS optimization;

2) In MSS optimization, parameters are updated by Singular Value Thresholding (SVT) method, while DLRL updates trainable weights and parameters with back propagation, according to a predefined reconstruction loss function;

3) MSS optimization solves the Schatten-p norm minimization problem upon singular values of matrix, while DLRL minimizes the norm values with learning steps based on matrix multiplication. This significantly accelerates the model computation as we have analyzed.

IV. Experimental Analyses

In this section, we consider low-rank matrix completion in collaborative filtering and image recovery as concrete examples to validate the efficiency of the proposed framework. Substantial experiments are conducted including the comparison with other related state-of-the-art methods. The parameter sensitivity, runtime, as well as convergence are analyzed. The proposed DLRL is implemented with PyTorch platform and run on the computer with an I5-7200U CPU and 8G RAM.

A. Dataset Descriptions

For collaborative filtering, we adopt six widely used datasets that are publicly available. These datasets are collected from practical applications like movie websites, shopping platforms, and music recommendation websites. The detailed information of these datasets is listed as follows:

Movielens[1] is a collection of widely used benchmark datasets for ratings on movies. In experiments, we select Movielens-100K and Movielens-1M which contain 100 thousand ratings and 1 million ratings, respectively.

Filmtrust[2] is a dataset for movie recommendation that was collected from the well-known Filmtrust website. The dataset has 35,497 ratings generated by 1,508 users and 2,071 items from the website.

Netflix[3] is a popular online movie and TV website which contains about 100 million ratings. Limited by computational resources and time complexity of compared methods, we only extract part of the data with 1,500 users and 2,000 items.

Amazon Music[4] is a famous 5-star music recommendation database collected from the Amazon website. We generate the dataset by extracting users who rated at least 30 items and items that were rated by at least 60 users.

Epinions[5] is a product recommendation dataset which records 5-star ratings of items. In this paper, we select all ratings produced by the top 4,000 users and 12,000 items.

Statistics of these practical recommendation datasets are shown in Table I. It can be seen from the table that all of these datasets are sparse, which brings challenges for matrix completion. We set 80% observed ratings as the training set Ω while the others as the testing set $\bar{\Omega}$.

As to image recovery tasks, several images of three channels are selected, where the size of each image is $300 \times 300 \times 3$ and we will conduct matrix completion on each channel (i.e., red, green, and blue). We generate incomplete images with random masks or text masks as shown in Figure 2, and recover the missing pixel values.

[1]https://grouplens.org/datasets/movielens/
[2]https://www.librec.net/datasets.html
[3]https://www.kaggle.com/netflix-inc/netflix-prize-data
[4]http://jmcauley.ucsd.edu/data/amazon/
[5]http://www.trustlet.org/wiki/Epinions_dataset

Datasets	Filmtrust	Netflix	Amazon Music	MovieLens-100K	MovieLens-1M	Epinions
# of users	1,508	1,500	5,249	943	6,040	4,000
# of items	2,071	2,000	4,874	1,682	3,952	12,000
# of ratings	35,497	137,962	53,316	100,000	1,000,201	81,513
Rating scale	[0.5, 4.0]	[1.0, 5.0]	[1.0, 5.0]	[1.0, 5.0]	[1.0, 5.0]	[1.0, 5.0]
Data density	0.01137	0.04599	0.00208	0.06305	0.04190	0.00170

TABLE I: Statistics of the real-world datasets of collaborative filtering used in experiments.

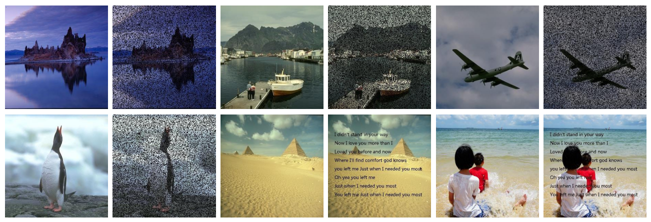

Fig. 2: Images (a) - (f) without/with random or text masks.

B. Performance Evaluation

To evaluate the performance of all compared methods, we adopt Root Mean Square Error (RMSE) defined by

$$\mathrm{RMSE} = \sqrt{\frac{1}{|\bar{\Omega}|} \sum_{(i,j)\in\bar{\Omega}} (x_{ij} - \hat{x}_{ij})} \tag{48}$$

to measure the reconstruction error of testing set $\bar{\Omega}$ in recommender system datasets, where $\hat{x}_{ij}$ and x_{ij} are entries in predicted matrix $\hat{\mathbf{X}}$ and observed matrix $\mathbf{X}$. A lower RMSE value indicates better performance.

The recovered images are evaluated by the widely employed Peak Signal-to-Noise Ratio (PSNR), defined as

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{255^2}{\frac{1}{3mn}\sum_{t=1}^{3}\|P_{\bar{\Omega}}(\hat{\mathbf{X}}^{(t)}) - P_{\bar{\Omega}}(\mathbf{X}^{(t)})\|_F^2}\right), \tag{49}$$

where $\hat{\mathbf{X}}^{(t)}$ and $\mathbf{X}^{(t)}$ are the recovered image matrix and original image matrix of the t-th channel, respectively. We evaluate the reconstruction error of unobserved pixels on all channels. A higher PSNR value corresponds to better quality of recovered images.

C. Compared Methods and Parameter Settings

To better evaluate the performance of the proposed method, we select six state-of-the-art methods on low-rank matrix recovery, including TNNR [15], IRNN [53], GPG [22], MSS [28], sRGCNN [54], DNNR [55], FNNM [56] and ISVTA [27]. All of these methods except FNNM and sRGCNN minimize the rank of the matrix by optimizing Equation (2), where $g(\mathbf{X})$ is the nonconvex non-smooth rank function. FNNM is specifically developed to minimize values of nuclear norm. Most of these methods include SVD computation at each iteration. Due to the limitation of FNNM and sRGCNN, we only apply them to collaborative filtering.

Some parameter settings are clarified in advance to achieve more credible experimental results. All algorithms apply default settings as original papers. TNNR with Accelerated Proximal Gradient Line (APGL) search method is selected, whose parameters are set as $\rho = 0.005$ and $\lambda = 0.01$. In IRNN, GPG and DNNR, we initialize $\lambda_0 = \alpha\|P_\Omega(\mathbf{X})\|_\infty$, where α ranges in $\{1, 100, 200, \cdots, 1000\}$. We select ℓ_p-norm as nonconvex surrogate function for IRNN and GPG, where p ranges in $(0, 1)$, while for DNNR we follow original paper and set $p = \frac{1}{2}$ or $\frac{2}{3}$. As for MSS, we set $\eta = 0.1$ and $\lambda = 200$, the factor number is set as 4 or 5 with $p_i = 1, i = 1, \cdots, I$, that is, $p = \frac{1}{4}$ or $\frac{1}{5}$. For modified Schatten-p norm optimization of ISVTA, we adopt $p = 0.1$.

As to the proposed DLRL, the dimensions of factors are set as $\{d_i\}_{i=1}^{I-1} = d$. Specifically, we set $d = 20$ for experiments on synthetic and recommendation datasets, while $d = 80$ for image recovery tasks. In following experiments, we set all $p_i = 1$ to construct different global p values in light of Equation (8). The learning rate of DLRL is set as 0.01 or 0.005.

D. Experiments on Synthetic Data

First, experiments on synthetic data are conducted to explore the parameter sensitivity of DLRL. The synthetic matrix $\mathbf{X}_s \in \mathbb{R}^{100\times100}$ used in experiments is generated by $\mathbf{X}_s = \mathbf{U}\mathbf{V}^T$, where $\mathbf{U} \in \mathbb{R}^{100\times20}$ and $\mathbf{V} \in \mathbb{R}^{100\times20}$. To evaluate the ability to recover corrupt matrix, we apply the Gaussian noisy $\mathcal{N}(0,\sigma)$ to every entry of the matrix, and set n_r as the percentage of unobserved data entries. We use the Relative Square Root Error (RSRE) defined as $\mathrm{RSRE} = \frac{\|\hat{\mathbf{X}}-\mathbf{X}_s\|_F}{\|\mathbf{X}_s\|_F}$ to measure the recovery accuracy, where a lower value of RSRE indicates better performance. Because the global p value of DLRL is tightly related to the selection of I and $\{p_i\}_{i=1}^{I}$, we directly evaluate the performance of DLRL with varying p values. For simplicity, we set $p_i = 1$ for $i = 1, \cdots, I$. Thus, different block numbers $I = \{10, 5, 4, 3, 2\}$ correspond to varying $p = \{\frac{1}{10}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}\}$, respectively. Figure 3 records the performance of compared methods with varying p values, noise magnitude σ and unobserved data percentage n_r. Because DNNR is only for $p = \frac{1}{2}$ and $p = \frac{2}{3}$, we plot RSRE values across different p values for comparison with other methods. From Figure 3, we observe that values of RSRE obtained by all compared methods become slightly worse with the increase of σ and n_r. The proposed DLRL is more robust to the change of p value when $p \leq \frac{1}{2}$, and generally achieves superior performance compared with other methods. When $p = \frac{1}{2}$, we have examined DLRL with $\{p_i = 1\}_{i=1}^{2}$ (two blocks) and $\{p_i = \frac{3}{2}\}_{i=1}^{3}$ (three blocks), finding that the performance is similar and DLRL with three blocks works better. Here we record the performance of DLRL with $\{p_i = \frac{3}{2}\}_{i=1}^{3}$. In summary, it is verified that lower p

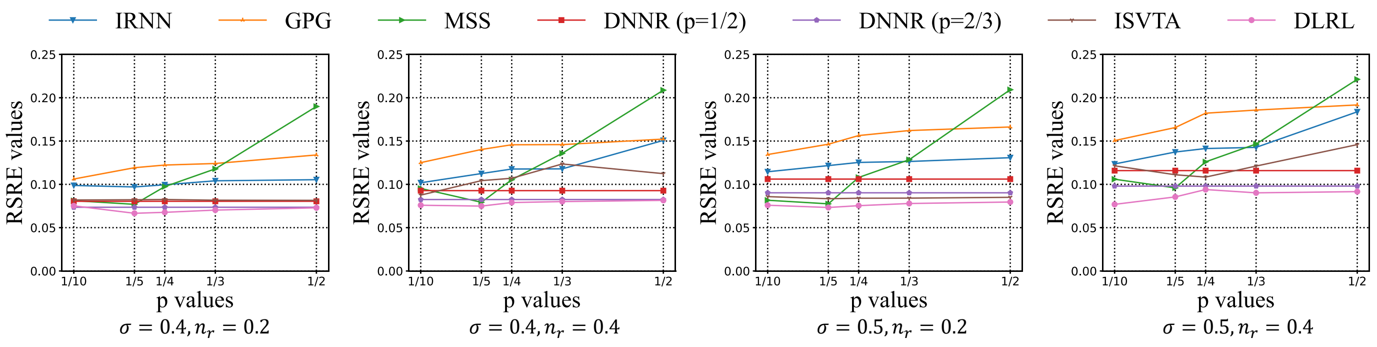

Fig. 3: Performance of compared methods (IRNN, GPG, MSS, DNNR, ISVTA and DLRL) on synthetic data with varying p values, noise magnitude σ and unobserved data percentage n_r. For DLRL, we set $p_i = 1$ for $i = 1, \cdots, I$, because of which different global p values shown in the x-axes can be approximated by varying selections of block number I.

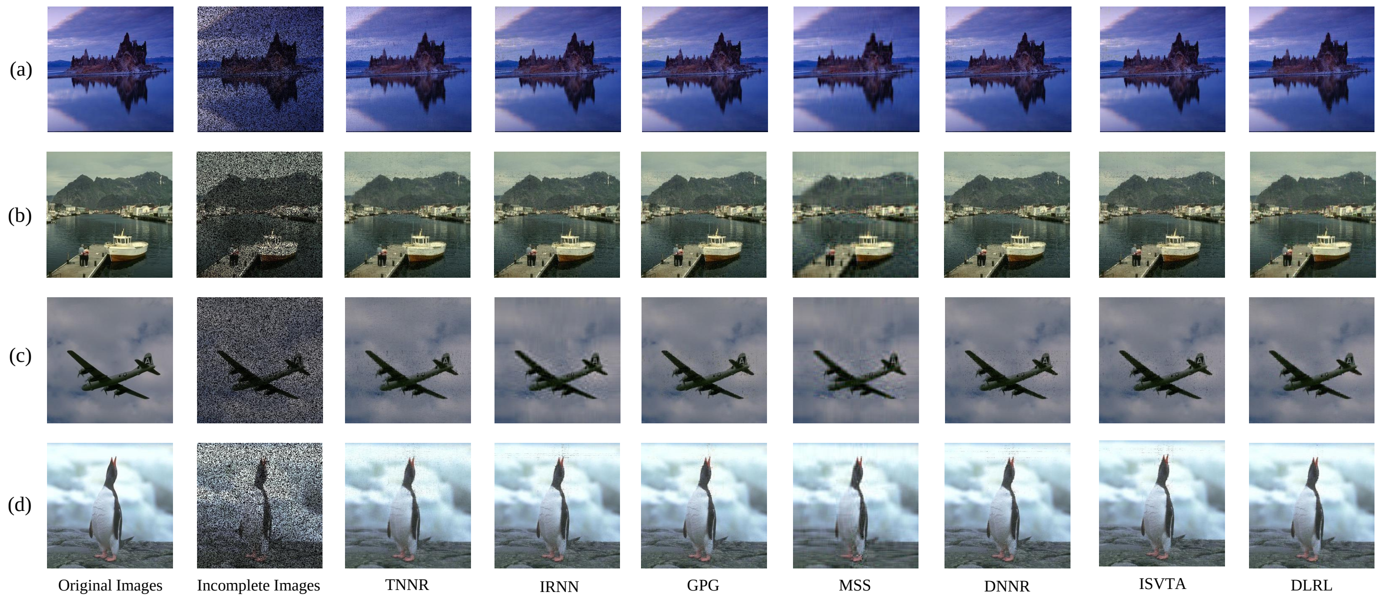

Fig. 4: Image recovery results generated by TNNR, IRNN, GPG, MSS, DNNR, ISVTA and DLRL, where the first column lists original images and the second column lists incomplete images.

values lead to better recovery performance for the proposed method, and DLRL with $p = \frac{1}{5}$ corresponds to favorable results in most cases. In addition, DLRL with $p = \frac{1}{10}$ may result in performance decline. These observations motivate us to set $p = \frac{1}{5}$ with all $p_i = 1$ in the subsequent experiments.

E. Application to Image Recovery

In this subsection, we consider two types of masks to generate incomplete images: random masks covering 40% pixels and text masks, and all methods need to fill the missing values of the incomplete matrices. Because a color image contains three channels, we run matrix completion methods on each channel respectively, and combine all 2-D matrices to generate final recovery results.

Figure 4 shows recovery results of Images (a) - (d) yielded by all low-rank matrix recovery methods. From the figure, we observe that most methods perform well when recovering incomplete images generated by random masks. However, as to the recovery of Images (e) - (f) corrupted by text mask shown in Figures 5 and 6, TNNR, IRNN, and GPG fail to remove the text completely. It follows from Table II that the proposed DLRL gains higher PSNR on all test images. Compared with MSS which also applies multi-Schatten-p norm surrogate function, the proposed DLRL achieves significant improvement on some tested images.

F. Application to Collaborative Filtering

To further examine the feasibility of the proposed method on larger datasets, we conduct substantial experiments on datasets for collaborative filtering. In this subsection, all methods execute matrix completion tasks on incomplete user-item rating matrices of recommender systems. Table III and Figure 7 record the RMSE values and runtimes of all compared algorithms on test datasets, respectively. From the experimental results, we have the following beneficial observations. The proposed DLRL performs the best on five of six test datasets and significantly promotes the performance, which is also competitive on Netlifx dataset. Although sRGCNN obtains the best accuracy on Netflix dataset, its computational cost is drastically high. The experimental results reveal that MSS and

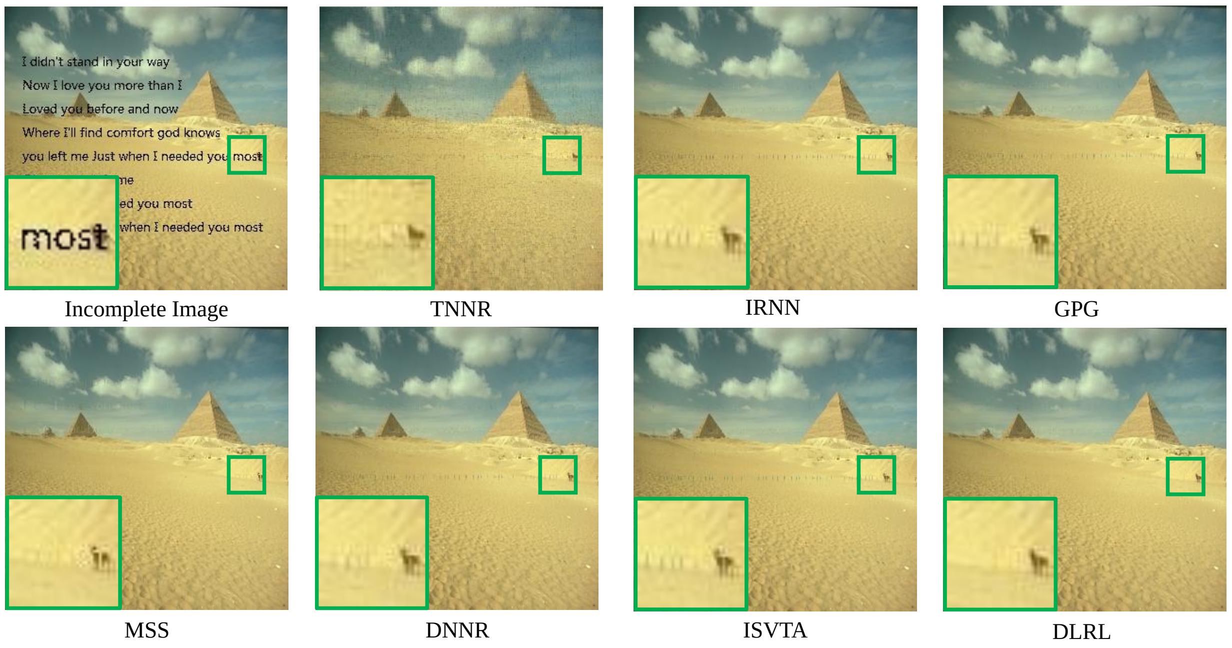
I didn't stand in your way
Now I love you more than I
Loved you before and now
Where I'll find comfort god knows
you left me Just when I needed you most
me
ed you most
when I needed you most
most
Incomplete Image
TNNR
IRNN
GPG
MSS
DNNR
ISVTA
DLRL

Fig. 5: Image recovery results of corrupted Image (e) generated by TNNR, IRNN, GPG, MSS, DNNR, ISVTA and DLRL.

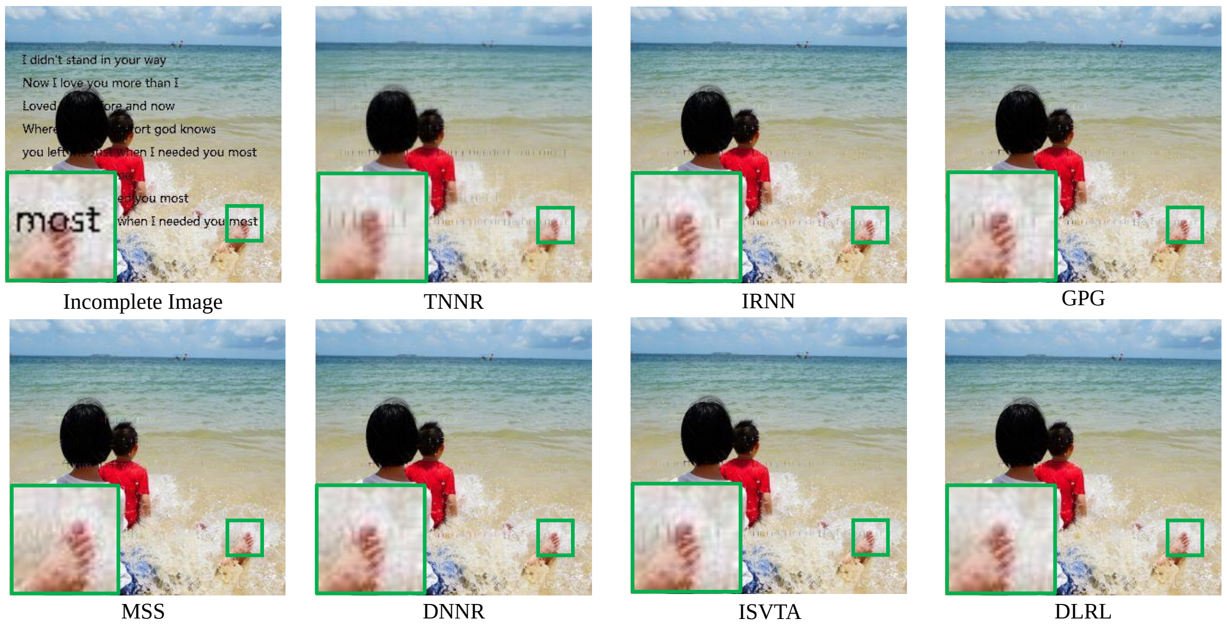

Fig. 6: Image recovery results of corrupted Image (f) generated by TNNR, IRNN, GPG, MSS, DNNR, ISVTA and DLRL.

Methods/Images	a	b	c	d	e	f
TNNR	25.71 (0.21)	22.67 (0.28)	26.43 (0.21)	25.74 (0.17)	25.48 (0.36)	21.81 (0.41)
IRNN	26.81 (0.23)	23.22 (0.21)	28.80 (0.23)	27.88 (0.22)	27.48 (0.33)	21.81 (0.39)
GPG	27.03 (0.17)	23.06 (0.16)	28.58 (0.24)	28.15 (0.17)	27.09 (0.39)	23.54 (0.44)
MSS	29.66 (0.18)	23.45 (0.21)	28.79 (0.31)	27.84 (0.15)	27.48 (0.30)	21.75 (0.45)
DNNR	28.86 (0.21)	22.64 (0.25)	28.65 (0.26)	28.30 (0.19)	27.70 (0.32)	25.50 (0.46)
ISVTA	29.11 (0.24)	22.84 (0.27)	29.08 (0.24)	28.42 (0.16)	27.31 (0.28)	24.47 (0.33)
DLRL	**29.79 (0.16)**	**23.68 (0.22)**	**29.57 (0.19)**	**29.09 (0.20)**	**28.20 (0.35)**	**25.87 (0.40)**

TABLE II: PSNR values (mean and standard deviation) of all compared methods on image recovery shown in Figures 4 - 6, where the best performance is highlighted in bold. All experiments are run 10 times and we record the average PSNR values and standard deviations.

Methods/Datasets	Filmtrust	Netflix	Amazon Music	MovieLens-100K	MovieLens-1M	Epinions
TNNR	1.1617 (0.82%)	1.0134 (0.35%)	- (-%)	0.9878 (0.25%)	- (-%)	- (-%)
IRNN	1.0016 (1.01%)	0.9796 (0.23%)	1.7157 (0.98%)	0.9387 (0.44%)	0.9304 (0.68%)	1.9780 (0.95%)
GPG	1.0194 (1.13%)	0.9452 (0.29%)	1.9862 (1.43%)	0.9489 (0.31%)	0.9341 (0.45%)	1.6669 (0.84%)
MSS	0.8106 (1.08%)	0.9496 (0.38%)	0.9187 (0.86%)	0.9505 (0.28%)	0.8719 (0.33%)	1.1671 (0.71%)
sRGCNN	0.7964 (1.11%)	**0.8984 (0.37%)**	0.9214 (0.78%)	0.9271 (0.33%)	0.8815 (0.42%)	1.1923 (0.88%)
DNNR	0.9425 (0.94%)	0.9629 (0.26%)	1.6481 (1.21%)	0.9507 (0.41%)	0.9716 (0.49%)	1.7157 (0.92%)
FNNM	1.1948 (1.23%)	0.9697 (0.31%)	1.8845 (1.06%)	0.9935 (0.37%)	0.9691 (0.51%)	1.8324 (0.88%)
ISVTA	1.1095 (0.97%)	1.0111 (0.29%)	1.8157 (1.24%)	0.9846 (0.31%)	0.9349 (0.42%)	1.7649 (0.92%)
DLRL	**0.7429 (0.87%)**	0.9083 (0.33%)	**0.8822 (0.87%)**	**0.8892 (0.20%)**	**0.8315 (0.41%)**	**1.1587 (0.78%)**

TABLE III: RMSE values (mean and standard deviation%) of all compared methods on all collaborative filtering datasets, where the best performance is highlighted in bold. All experiments are run 10 times and we record the average RMSE values and standard deviations. Limited by computation resources and high time complexity, we only run TNNR on datasets with low dimension.

the proposed DLRL, which are based on multi-factor strategy, generally gain lower RMSE values on most datasets. On Amazon music and Epinions datasets that are extremely sparse, all of TNNR, IRNN, GPG, DNNR, FNNM and ISVTA achieve undesirable performance, revealing that the DLRL framework behaves favorably on sparse datasets. Namely, DLRL is more effective in addressing the cold-start issue in recommender systems. In addition, all methods except MSS and DLRL require a much higher time consumption. While the runtime of DLRL is further lower than other methods, due to no SVD at each iteration. The runtime of MSS is also acceptable because it only conducts SVD of partial matrices instead of the entire matrix. Overall, the proposed DLRL improves the performance of MSS by both time consumption and recovery accuracy. One most possible reason for the accuracy improvement is that our algorithm calculates the gradient during back propagation based on the additionally defined reconstruction loss function, that is, Equation (26), which makes the model more inclined to yield an output with lower reconstruction error during training. Besides, the improvement may also be due to the fact that optimization target (15) is a strict version of the original optimization problem (10).

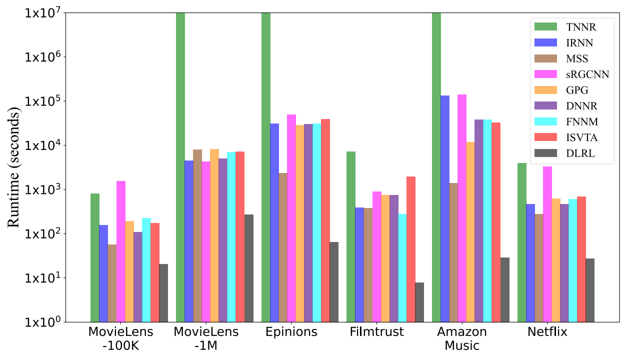

Fig. 7: Runtimes (seconds) comparison for all methods on all test datasets. Because the time cost for TNNR is unacceptable on Amazon Music, MovieLens-1M, and Epinions datasets, we plot the runtimes as 1×10^7 seconds in these cases.

For the purpose of investigating the effect of updating rate η, we examine the performance of DLRL with varying fixed η values in Figure 8. Table IV demonstrates the performance of DLRL with learnable η and fixed $\eta = 1$. We have some interesting observations through these experimental results. First, although the time consumption of DLRL with learnable η is not the lowest, it achieves superior accuracy on all test datasets, which is more significant on Filmtrust, Amazon Music, MovieLens-100K and Epinions datasets. Notice that we do not want an η which has the lowest runtime, but want to learn an η which can obtain the best accuracy with relatively less time consumption. The runtime of DLRL with learnable η is much less than that with fixed $\eta = 1$, which shows that DLRL with learnable η is effective. Second, according to Equation (24), a smaller η value should correspond to a faster updating rate of $\mathbf{Y}_i$. However, the experimental results indicate that the framework with lower fixed η does not always consume the least time. On the contrary, it requires more time on some datasets when η decreases to 0.1 or 0.01. This may be due to the excessively high updating rate which makes the algorithm fall into a suboptimal solution, and the optimizer has to consume more epochs to seek better values of trainable variables. On most datasets, the runtime decreases at first as the η decreases, and begins to increase after a certain point. As to the accuracy, RMSE value fluctuates marginally as η changes. We also test an even lower $\eta < 0.1$ on Filmtrust and Epinions datasets, finding that the runtime and RMSE are higher than those with fixed $\eta = 0.1$. When η drops to 0.01, the results are even unacceptable on some datasets. It is noted that DLRL achieves its best performance on both runtime and RMSE when fixed $\eta = 0.07$ on Amazon Music dataset. Consequently, the selections of η are varied across different datasets, which indicates that an automatically learned η is more applicable to DLRL. Third, associated with Table IV, we can find that the learned η in DLRL which applies trainable η is basically close to the optimal fixed η that corresponds to the lowest RMSE in our experiments. The learned η is more accurate than the optimal fixed η, which validates that the learnable η is meaningful.

Figure 9 records values of the loss function and Figure 10 shows the sum of Schatten-p norm for all factors during training on all test collaborative filtering datasets, with varying learning rates. Because the initial loss values in some datasets are extremely large, we record the logarithm loss values for better readability. It can be seen from Figure 9 that loss function values decrease quickly, which is one of the reasons that the presented method runs faster than compared methods. Owing to the fact that we apply the learning rate decay strategy during training, the curves of loss values may fluctuate or continue to decline after a period of stable loss values.

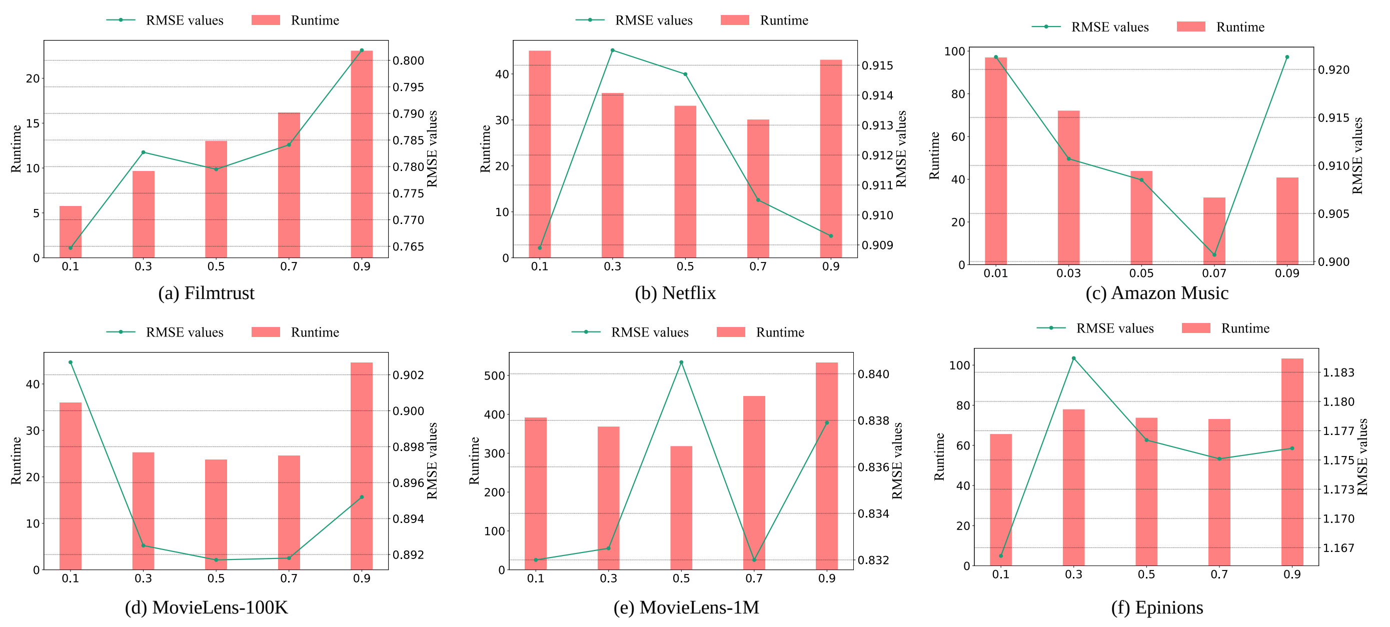

Fig. 8: Performance (RMSE values and runtime) of DLRL with various fixed η values. Specially, because the optimal accuracy on Amazon Music is obtained at $\eta = 0.07$, we show the experimental results with fixed η ranging in $\{0.01, 0.03, 0.05, 0.07, 0.09\}$.

η	Metrics	Filmtrust	Netflix	Amazon Music	MovieLens-100K	MovieLens-1M	Epinions
$\eta = 1.0$	Runtime (s)	30.11	52.76	90.21	60.43	713.84	186.43
	RMSE	0.8122 (0.891%)	0.9097 (0.31%)	0.9429 (0.79%)	0.8894 (0.23%)	0.8383 (0.46%)	1.1755 (0.77%)
Learnable η	Runtime (s)	7.93	27.18	29.26	20.64	270.53	64.78
	RMSE	**0.7429 (0.87%)**	**0.9083 (0.33%)**	**0.8822 (0.87%)**	**0.8892 (0.20%)**	**0.8315 (0.41%)**	**1.1587 (0.78%)**
	Learned η	0.1813	0.1270	0.0775	0.5316	0.2364	0.1153

TABLE IV: Performance (runtime and RMSE) of DLRL with fixed $\eta = 1.0$ and adaptive learned η values, where the best accuracy is highlighted in bold.

Meanwhile, as the training going, the Schatten-p norm values of factors decrease monotonically, which is consistent with previous theoretical analyses. It can be seen that different selections of learning rates may lead to varied suboptimal solutions. These observations point out that the proposed method succeeds in minimizing the rank of matrices and reconstruction errors at the same time.

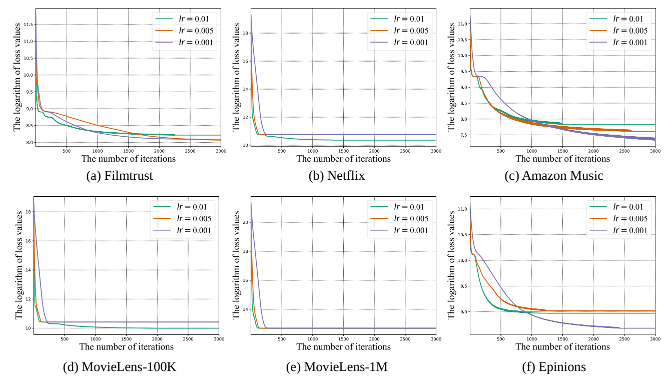

Fig. 9: Convergence curves of logarithm loss values with varying learning rates lr ranging in $\{0.01, 0.005, 0.001\}$ on all tested collaborative filtering datasets.

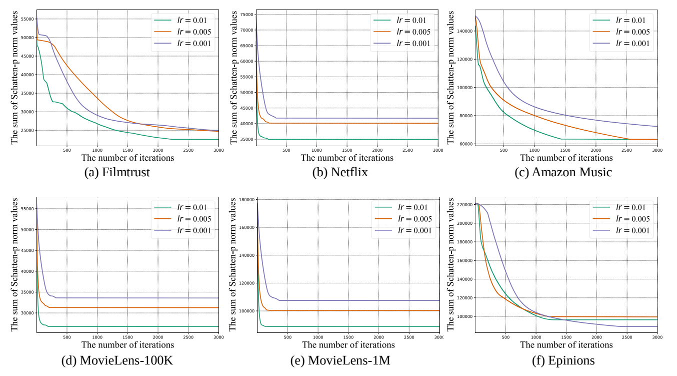

Fig. 10: The sum of Schatten-p norm values for all factors $\mathbf{X}_i, i = 1, \cdots, 5$ with varying learning rates lr ranging in $\{0.01, 0.005, 0.001\}$ on all tested collaborative filtering datasets.

V. Conclusion

In this paper, we proposed an end-to-end differentiable low-rank learning framework for learning low-rank optimization problem with MSS function, which solved nonconvex and discontinuous rank optimization problem efficiently. The proposed method was applied to low-rank matrix completion problems and we expanded the optimization target with the gradient proximal mapping method, which was reformulated with the Lagrangian multiplier method and transformed into a network structure. The proposed framework avoided conducting SVD during iterations and further accelerated the computation speed. Finally, experimental results verified that the proposed method succeeded in solving low-rank matrix completion problems with applications to image recovery and collaborative filtering, achieving encouraging performance by both accuracy and computational cost. In the future, we will further consider the potential effects of solving nondifferentiable low-rank problems with neural network-based methods.

REFERENCES

[1] Y. Du, M. Fang, J. Yi, C. Xu, J. Cheng, and D. Tao, "Enhancing the robustness of neural collaborative filtering systems under malicious attacks," *IEEE Transactions on Multimedia*, vol. 21, no. 3, pp. 555–565, 2019.

[2] L. T. Nguyen and B. Shim, "Low-rank matrix completion using graph neural network," in *Proceedings of International Conference on Information and Communication Technology Convergence*, pp. 17–21, 2020.

[3] Y. Liu, Z. Long, and C. Zhu, "Image completion using low tensor tree rank and total variation minimization," *IEEE Transactions on Multimedia*, vol. 21, no. 2, pp. 338–350, 2019.

[4] J. Mu, R. Xiong, X. Fan, D. Liu, F. Wu, and W. Gao, "Graph-based non-convex low-rank regularization for image compression artifact reduction," *IEEE Transactions on Image Processing*, vol. 29, pp. 5374–5385, 2020.

[5] X. Xiao, Y. Chen, Y.-J. Gong, and Y. Zhou, "Low-rank preserving t-linear projection for robust image feature extraction," *IEEE Transactions on Image Processing*, vol. 30, pp. 108–120, 2021.

[6] Y. Han, J. Liu, and X. Luo, "An improved latent low rank representation for automatic subspace clustering," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 5188–5189, 2020.

[7] Z. Wang, L. Wang, J. Wan, and H. Huang, "Shared low-rank correlation embedding for multiple feature fusion," *IEEE Transactions on Multimedia*, 2020.

[8] P. Jing, Y. Su, L. Nie, X. Bai, J. Liu, and M. Wang, "Low-rank multi-view embedding learning for micro-video popularity prediction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 8, pp. 1519–1532, 2018.

[9] P. Jing, J. Zhang, L. Nie, S. Ye, J. Liu, and Y. Su, "Tripartite graph regularized latent low-rank representation for fashion compatibility prediction," *IEEE Transactions on Multimedia*, 2021.

[10] L. Zhang, J. Yin, P. Li, Y. Shang, R. Zimmermann, and L. Shao, "Flickr image community analytics by deep noise-refined matrix factorization," *IEEE Transactions on Multimedia*, vol. 22, no. 5, pp. 1273–1284, 2020.

[11] J. Chen, C. Wang, S. Zhou, Q. Shi, J. Chen, Y. Feng, and C. Chen, "Fast adaptively weighted matrix factorization for recommendation with implicit feedback," in *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pp. 3470–3477, AAAI Press, 2020.

[12] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM Review*, vol. 52, no. 3, pp. 471–501, 2010.

[13] Y. Huang, G. Liao, Y. Xiang, L. Zhang, J. Li, and A. Nehorai, "Low-rank approximation via generalized reweighted iterative nuclear and frobenius norms," *IEEE Transactions on Image Processing*, vol. 29, pp. 2244–2257, 2020.

[14] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan, "Tensor robust principal component analysis with a new tensor nuclear norm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 4, pp. 925–938, 2020.

[15] Y. Hu, D. Zhang, J. Ye, X. Li, and X. He, "Fast and accurate matrix completion via truncated nuclear norm regularization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 9, pp. 2117–2130, 2013.

[16] J. Xu, L. Zhang, D. Zhang, and X. Feng, "Multi-channel weighted nuclear norm minimization for real color image denoising," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1096–1104, 2017.

[17] X. Lan, S. Zhang, P. C. Yuen, and R. Chellappa, "Learning common and feature-specific patterns: a novel multiple-sparse-representation-based tracker," *IEEE Transactions on Image Processing*, vol. 27, no. 4, pp. 2022–2037, 2017.

[18] C. Lu, J. Tang, S. Yan, and Z. Lin, "Nonconvex nonsmooth low rank minimization via iteratively reweighted nuclear norm," *IEEE Transactions on Image Processing*, vol. 25, no. 2, pp. 829–839, 2016.

[19] J. H. Friedman, "Fast sparse regression and classification," *International Journal of Forecasting*, vol. 28, no. 3, pp. 722–738, 2012.

[20] C. Zhang *et al.*, "Nearly unbiased variable selection under minimax concave penalty," *The Annals of Statistics*, vol. 38, no. 2, pp. 894–942, 2010.

[21] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle properties," *Journal of the American Statistical Association*, vol. 96, no. 456, pp. 1348–1360, 2001.

[22] C. Lu, C. Zhu, C. Xu, S. Yan, and Z. Lin, "Generalized singular value thresholding," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 1805–1811, 2015.

[23] T. H. Oh, Y. Tai, J. Bazin, H. Kim, and I. S. Kweon, "Partial sum minimization of singular values in robust PCA: algorithm and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 4, pp. 744–758, 2016.

[24] S. Gu, Q. Xie, D. Meng, W. Zuo, X. Feng, and L. Zhang, "Weighted nuclear norm minimization and its applications to low level vision," *International Journal of Computer Vision*, vol. 121, no. 2, pp. 183–208, 2017.

[25] Y. Xie, S. Gu, Y. Liu, W. Zuo, W. Zhang, and L. Zhang, "Weighted schatten p-norm minimization for image denoising and background subtraction," *IEEE Transactions on Image Processing*, vol. 25, no. 10, pp. 4842–4857, 2016.

[26] X. Huang, B. Du, and W. Liu, "Multichannel color image denoising via weighted schatten p-norm minimization," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 637–644, 2020.

[27] H. Zhang, J. Qian, B. Zhang, J. Yang, C. Gong, and Y. Wei, "Low-rank matrix recovery via modified schatten-p norm minimization with convergence guarantees," *IEEE Transactions on Image Processing*, vol. 29, pp. 3132–3142, 2020.

[28] C. Xu, Z. Lin, and H. Zha, "A unified convex surrogate for the schatten-p norm," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pp. 926–932, 2017.

[29] F. Nie, H. Huang, and C. H. Q. Ding, "Low-rank matrix recovery via efficient schatten p-norm minimization," in *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[30] M. Zhang, Z. Huang, and Y. Zhang, "Restricted p -isometry properties of nonconvex matrix recovery," *IEEE Transactions on Information Theory*, vol. 59, no. 7, pp. 4316–4323, 2013.

[31] S. Arora, N. Cohen, W. Hu, and Y. Luo, "Implicit regularization in deep matrix factorization," in *Advances in Neural Information Processing Systems*, pp. 7413–7424, 2019.

[32] S. Yang, L. Li, S. Wang, W. Zhang, Q. Huang, and Q. Tian, "Skeletonnet: A hybrid network with a skeleton-embedding process for multi-view image representation learning," *IEEE Transactions on Multimedia*, vol. 21, no. 11, pp. 2916–2929, 2019.

[33] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proceedings of the Twenty-Seventh International Conference on Machine Learning*, pp. 399–406, 2010.

[34] J. Zhang and B. Ghanem, "Ista-net: Interpretable optimization-inspired deep network for image compressive sensing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1828–1837, 2018.

[35] Y. Yang, J. Sun, H. Li, and Z. Xu, "Deep admm-net for compressive sensing MRI," in *Advances in Neural Information Processing Systems*, pp. 10–18, 2016.

[36] S. Du, Z. Liu, Z. Chen, W. Yang, and S. Wang, "Differentiable bi-sparse multi-view co-clustering," *IEEE Transactions on Signal Processing*, vol. 69, pp. 4623–4636, 2021.

[37] X. Xie, J. Wu, G. Liu, Z. Zhong, and Z. Lin, "Differentiable linearized ADMM," in *Proceedings of the Thirty-Sixth International Conference on Machine Learning*, vol. 97, pp. 6902–6911, 2019.

[38] S. Wang, Z. Chen, S. Du, and Z. Lin, "Learning deep sparse regularizers with applications to multi-view clustering and semi-supervised classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[39] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems*, pp. 556–562, 2001.

[40] Y. Lu, C. Yuan, W. Zhu, and X. Li, "Structurally incoherent low-rank nonnegative matrix factorization for image classification," *IEEE Transactions Image Processing*, vol. 27, no. 11, pp. 5248–5260, 2018.

[41] J. Wang, F. Tian, H. Yu, C. H. Liu, K. Zhan, and X. Wang, "Diverse non-negative matrix factorization for multiview data representation," *IEEE Transactions on Cybernetics*, vol. 48, no. 9, pp. 2620–2632, 2018.
[42] H. Wang, Y. Cen, Z. He, R. Zhao, Y. Cen, and F. Zhang, "Robust generalized low-rank decomposition of multimatrices for image recovery," *IEEE Transactions on Multimedia*, vol. 19, no. 5, pp. 969–983, 2017.
[43] A. Acharya, R. Goel, A. Metallinou, and I. S. Dhillon, "Online embedding compression for text classification using low rank matrix factorization," in *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, pp. 6196–6203, 2019.
[44] A. Nitanda, "Stochastic proximal gradient descent with acceleration techniques," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1574–1582, 2014.
[45] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Scalable collaborative filtering approaches for large recommender systems," *Journal of Machine Learning Research*, vol. 10, pp. 623–656, 2009.
[46] R. Mazumder, T. Hastie, and R. Tibshirani, "Spectral regularization algorithms for learning large incomplete matrices," *The Journal of Machine Learning Research*, vol. 11, pp. 2287–2322, 2010.
[47] F. Shang, Y. Liu, and J. Cheng, "Scalable algorithms for tractable schatten quasi-norm minimization," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pp. 2016–2022, 2016.
[48] F. Shang, Y. Liu, and J. Cheng, "Tractable and scalable schatten quasi-norm approximations for rank minimization," in *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Statistics*, vol. 51, pp. 620–629, 2016.
[49] F. Shang, J. Cheng, Y. Liu, Z. Luo, and Z. Lin, "Bilinear factor matrix norm minimization for robust PCA: algorithms and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 9, pp. 2066–2080, 2018.
[50] J. Bolte, S. Sabach, and M. Teboulle, "Proximal alternating linearized minimization for nonconvex and nonsmooth problems," *Mathematical Programming*, vol. 146, no. 1-2, pp. 459–494, 2014.
[51] K.-C. Toh and S. Yun, "An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems," *Pacific Journal of optimization*, vol. 6, no. 615-640, p. 15, 2010.
[52] J. Bolte, S. Sabach, and M. Teboulle, "Proximal alternating linearized minimization for nonconvex and nonsmooth problems," *Mathematical Programming*, vol. 146, no. 1-2, pp. 459–494, 2014.
[53] C. Lu, J. Tang, S. Yan, and Z. Lin, "Generalized nonconvex nonsmooth low-rank minimization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4130–4137, 2014.
[54] F. Monti, M. M. Bronstein, and X. Bresson, "Geometric matrix completion with recurrent multi-graph neural networks," in *Advances in Neural Information Processing Systems*, pp. 3697–3707, 2017.
[55] H. Zhang, C. Gong, J. Qian, B. Zhang, C. Xu, and J. Yang, "Efficient recovery of low-rank matrix via double nonconvex nonsmooth rank minimization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 10, pp. 2916–2925, 2019.
[56] M. Yang, Y. Li, and J. Wang, "Feature and nuclear norm minimization for matrix completion," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

Zhaoliang Chen received his B.S. degree from the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China in 2019. He is currently pursuing the Ph.D. degree with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. His current research interests include machine learning, deep learning, graph neural networks and recommender systems.

Jie Yao received his B.S. degree from the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China in 2020. He is currently pursuing the M.S. degree with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. His main research interests include machine learning and computer vision.

Guobao Xiao received the B.S. degree in information and computing science from Fujian Normal University, China, in 2013, and the Ph.D. degree in computer science and technology from Xiamen University, China, in 2016. From 2016 to 2018, he was a Postdoctoral Fellow with the School of Aerospace Engineering, Xiamen University, China. He is currently a Full Professor with Minjiang University, China. He has published over 30 papers in the international journals and conferences. His research interests include machine learning, computer vision, pattern recognition, and bioinformatics.

Shiping Wang received his Ph.D. degree from the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China in 2014. He worked as a research fellow in Nanyang Technological University from August 2015 to August 2016. He is currently a professor with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. His research interests include machine learning, computer vision and granular computing.