Pattern Recognition 141 (2023) 109624

Contents lists available at ScienceDirect

Pattern Recognition

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

Tensor completion via convolutional sparse coding with small samples-based training

Tianchi Liao[a], Zhebin Wu[a], Chuan Chen[a,*], Zibin Zheng[b], Xiongjun Zhang[c]

[a] School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China
[b] School of Software Engineering, Sun Yat-sen University, Zhuhai 519000, China
[c] School of Mathematics and Statistics and Hubei Key Laboratory of Mathematical Sciences, Central China Normal University, Wuhan 430079, China

ARTICLE INFO

Article history:
Received 19 July 2022
Revised 26 March 2023
Accepted 21 April 2023
Available online 27 April 2023

Keywords:
Tensor completion
Convolutional sparse coding
High-pass filter
Inexact ADMM

ABSTRACT

Tensor data often suffer from missing value problems due to the complex high-dimensional structure while acquiring them. To complete the missing information, lots of Low-Rank Tensor Completion (LRTC) methods have been proposed, most of which depend on the low-rank property of tensor data. In this way, the low-rank component of the original data could be recovered roughly. However, the shortcoming is that the detailed information can not be fully restored, no matter the Sum of the Nuclear Norm (SNN) nor the Tensor Nuclear Norm (TNN) based methods. On the contrary, in the field of signal processing, Convolutional Sparse Coding (CSC) can provide a good representation of the high-frequency component of the image, which is generally associated with the detail component of the data. To this end, we propose two novel methods, LRTC-CSC-I and LRTC-CSC-II, which adopt CSC as a supplementary regularization for LRTC to capture the high-frequency components. Therefore, the LRTC-CSC methods can not only solve the missing value problem but also recover the details. Moreover, the regularizer CSC can be trained with small samples due to the sparsity characteristic. Extensive experiments show the effectiveness of LRTC-CSC methods, and quantitative evaluation indicates that the performance of our models are superior to state-of-the-art methods.

1. Introduction

A tensor is often known as an extension of a 1D vector or 2D matrix, which can offer a high-dimensional storage structure for various data nowadays, such as color images, hyperspectral images, videos, etc. Thus, it's widely leveraged in the field of computer vision, data mining [1], machine learning [2], and large-scale data analysis[3]. However, it's common that the obtained tensor data is nevertheless undersampled when processing it, which derives many Low-Rank Tensor Completion (LRTC) [4] algorithms to address the missing value problem. The general LRTC algorithm is formulated as:

$$\min_{\mathcal{X}} \quad \text{rank}(\mathcal{X}) \quad \text{s.t.} \quad \mathcal{X}_\Omega = \mathcal{T}_\Omega, \tag{1}$$

where $\mathcal{X}$ is the underlying tensor, $\mathcal{T}$ is the original data, Ω is the index set which implies the location corresponding to the observed entries. However, it's intractable to solve problem (1) which is NP-hard [5], due to the combinational nature of the function rank$(\cdot)$.

The definition of tensor rank is different from the known matrix's, because it is not unique. It depends on the tensor decomposition method used, e.g., CANDECOMP/PARAFAC (CP), Tucker, Tensor-Train (TT) etc. The definition of tensor rank, CP rank specifically, was first proposed by Kruskal *et al.* [6]. Immediately following, Tucker rank was adopted to approach the LRTC and gained unexpected achievements [4]. However, the drawback of Tucker rank is that its components will be the ranks of highly unbalanced flattened matrices if the original tensor is high dimensional (N>3). To address above problem, Bengua [7] proposed a novel LRTC solver based on TT rank. Besides, they employ the Ket Augmentation (KA) to extend low dimensional tensors to higher ones and proves that TT rank is more capable of global information capturing when the dimension of a tensor is larger than three [8]. Another novel definition of tensor rank named tubal rank is based on the recently proposed t-SVD decomposition [9], which is able to characterize the inherent low-rank structure of a tensor. Actually, no matter what rank is exploited to recover corrupted tensors, the core of these LRTC models is to find the relationship between missing entries and retained ones.

Despite the tensor rank estimation issue is intractable, nuclear norm [10] has been proved to be the most effective convex surrogate for the function rank$(\cdot)$ of the matrix. Liu *et al.* [4] proposed the sum of the nuclear norm (SNN) as a relaxation of the tensor rank.

* Corresponding author.
E-mail address: chenchuan@mail.sysu.edu.cn (C. Chen).

https://doi.org/10.1016/j.patcog.2023.109624
0031-3203/

In addition, the tensor nuclear norm (TNN) [11] is proposed from t-product to keep consistent with the matrix cases on concepts (see details in Section 2.1). No matter which nuclear norm is, the strong prior of the underlying tensor data is its low-rank property, and then the low-rank component can be recovered based on that.

Inevitably, the details of observation are overlooked by LRTC models. Thus, lots of regularizations are proposed to be employed as another prior [12] to improve the details of recovered data, e.g., Total Variation (TV) [13], framelets [14], wavelets [15] etc. Among these priors, TV is the most popular one in LRTC problems. Li *et al.* [13] combined TV and low-rankness to recover visual tensors and proposed a novel LRTC-TV model, where TV accounts for local piecewise priors. The starting point of LRTC-TV lies on the fact that the objects or edges in the spatial dimension will hold a smooth and piecewise structure. As a complementary prior to low rankness, TV can explore the local features properly. While Jiang *et al.* [16] combined TNN and an anisotropic TV to propose a TNN-3DTV model to exploit the correlations among the spatial and channel domains. The only fly in the ointment is that TV assumes the underlying tensor is piecewise smooth, resulting in undesired patch effects on the recovered tensors [17]. Therefore, Li *et al.* proposes a non-local self-similar completion model that incorporates a non-local prior to enhance the self-similarity of the underlying tensor [18]. What's more, the regularizations above are all only based on the information within that image, which is fatal when the remaining entries are few. Zhang *et al.* [19] utilized Convolutional Neural Networks (CNN) to train a set of effective denoisers to solving the image denoising problem, which shows the importance of feature collections. Thus, it's necessary to introduce other prior information outer the image for a better result.

Another prominent paradigm in signal processing is sparse coding. By seeking a sparse representation under an overcomplete dictionary, the underlying data could be restored exceptionally. The fundamental of sparse coding is the dictionary learning, a process to obtain a dictionary D that can best represent a set of train data. As early as 2006, Elad *et al.* [20] leveraged dictionary learning-based method to tackle the image denoising problem. Later in 2010, Yang *et al.* [21] decided to sparsely code for each patch of the low-resolution input and then used the coefficients of learned representation to generate the high-resolution output. In this way, the image super-resolution could be represented by sparse entries.

For a fixed dictionary $D \in \mathbb{R}^{N\times M}$, given a signal $X \in \mathbb{R}^N$, the task of seeking its sparsest representation $\Gamma \in \mathbb{R}^M$ is called sparse coding. Mathematically, the sparse coding problem can be formulated as:

$$\min_{\Gamma} \quad \|\Gamma\|_0 \quad \text{s.t.} \quad D\Gamma = X, \tag{2}$$

where $\|\Gamma\|_0$ denotes the number of non-zeros in Γ. There exists a convex relaxation of problem (2) in the form of Basis-Pursuit (BP) problem [22], formulated as:

$$\min_{\Gamma} \quad \|\Gamma\|_1 \quad \text{s.t.} \quad D\Gamma = X. \tag{3}$$

However, most traditional dictionary learning methods are patch-based, which might ignore the consistency among the entries lied in different patches but have some common features in the image. Besides, the learned data will contain shifted versions of the same features, resulting in an over-redundant dictionary.

To tackle the above problems, an alternative model, Convolutional Sparse Coding (CSC), has gained great attention in machine learning for image and video processing [23] recently due to its shift-invariance property. In 2010, Zeiler *et al.* [24] first proposed the concept of CSC. To improve the computional effectiveness, Bristow *et al.* [25] proposed novel and fast solutions under the Alternating Direction Method of Multipliers (ADMM) framework in succession. Furthermore, Wohlberg [26] demonstrated that suitable penalties on the gradients of the coefficient maps will improve the impulse noise denoising performance. Papyan *et al.* [27] introduced the relationship between CSC and CNN and claimed that CNN can be analyzed theoretically via multi-layer CSC.

In terms of applications, Papyan *et al.* [23] also proposed a slice-based dictionary learning algorithm, which was utilized in the CSC model to inpaint image as well as separate texture and cartoon. Recently, Zhang *et al.* [28] integrated the concept of low-rankness into CSC and addressed the rain streak removal. The work of Bao *et al.* [29] demonstrated that the CSC is able to process the high-frequency component of an image. In order to improve the capacity of the dictionary for learning multi-dimensional data, Bibi *et al.* [30] leveraged high order algebra to allow traditional CSC models encoding tensors. In addition, Xu *et al.* [31] decomposed the tensor dictionary and reconstructed it under the orthogonality-constrained convolutional factorization scheme to reduce computation costs. All in all, it can be argued that the success of CSC may attribute to the idea, "think globally and work locally".

Actually, the regularizations like TV, framelets, and wavelets only play the primary role in a single image, which can be treated as forcing a hard physics constraint on underlying data. The CNN-based regularization is often trained with thousands of data, which feature collections can explain in statistics. For example, the small CNN "FDNet" used in [32] was first proposed in [33], but it required more than five thousand images to train for more than two days. However, in the real world, it's not always available to acquire such a large amount of samples in a limited time. Therefore, the following facts motivate us to adopt CSC as a priori. Firstly, CSC can be valid for capturing and reconstructing high-frequency information, which is mainly reflected in the details and textures of images for image data. Secondly, due to the convenience of CSC, it can be trained with small samples (in this work we employed only 10 samples for training) and can extract the optimal representation dictionary in a limited training set, in which the CNN-based regularizations might be under-fitting.

Inspired by the capacity of CSC on extracting features locally with an overcomplete dictionary, two novel LRTC-CSC models are proposed to address the LRTC problem. By regarding CSC as a plug-and-play submodel in the optimization framework of the whole model, we effectively augment the high-frequency component of the underlying tensor. In this way, the global structure is handled by LRTC prior, and then CSC is leveraged to introduce external images as a priori to assist in preserving image details. Eventually, both global and local features can be well recovered. The main contributions of this paper are:

1. To tackle the missing value problem, we proposed two CSC regularized LRTC models, SNN-based and TNN-based, respectively. In both models, firstly, an overcomplete dictionary is pre-trained only with a minimal amount of data. And then the dictionary is used in CSC to restore the details of the underlying tensor. As a result, both low-rank components and details are well recovered.
2. We came up with effective algorithms to solve the LRTC-CSC-I and LRTC-CSC-II models, which are based on the inexact ADMM method [34] and plug-and-play framework [32]. With the variable splitting techniques, the entire problem can be split into three subproblems and solved separately.
3. We tested our model on different kinds of datasets, including color images, MRI data, and videos. Extensive experiments have verified the effectiveness of our model and claimed that the CSC regularization is suitable for different LRTC models. Furthermore, the performance of LRTC-CSC-II is superior to state-of-the-art models in the experiments.

2. Notations and preliminaries

2.1. Notations

Throughout this paper, the $(i_1 i_2 \cdots i_n)$-th element of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}$ is denoted by $x_{i_1 i_2 \cdots i_n}$.Vectors are denoted by boldface lowercase letters, e.g. $\boldsymbol{\alpha}$ and scalars by lowercase letters, e.g. α. We denote an N-order tensor by calligraphic letters, e.g., $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}$, where $I_k, k = 1, 2, \cdots, n$ is the dimension of k-th mode and $\mathbb{R}$ is the field of real number. Let the upper case latters, e.g., X, denote the matrices. Especially, a mode-k matricization (also known as mode-k unfolding or flattening) of a tensor $\mathcal{X}$ is reshaping the tensor into a matrix $X_{(k)} \in \mathbb{R}^{I_k \times (I_1 \cdots I_{k-1} I_{k+1} \cdots I_n)}$, which is defined as Tucker rank. For a 3-order tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, we denote its (i, j)-th mode-1, mode-2, and mode-3 fibers as $\mathcal{X}(:, i, j)$, $\mathcal{X}(i, :, j)$, and $\mathcal{X}(i, j, :)$. We use the Matlab notation $\mathcal{X}(i, :, :)$, $\mathcal{X}(:, i, :)$, and $\mathcal{X}(:, :, i)$ to denote the i-th horizontal, lateral and frontal slices, respectively. More often, $X^{(i)}$ and tube are used to represent $\mathcal{X}(:, :, i)$ and mode-3 fiber, respectively.

In addition, there are some mathematical operations of matrices and tensors used in this paper. The inner product of $X \in \mathbb{R}^{n \times m}$ and $Y \in \mathbb{R}^{n \times m}$ is defined as $\langle X, Y \rangle = tr(X^H Y)$, where X^H is the conjugate transpose of the matrix X and $tr(\cdot)$ is matrix trace. The l_1-norm is defined as $\|\mathcal{X}\|_1 = \sum_{i_1 i_2 \cdots i_n} |x_{i_1 i_2 \cdots i_n}|$, the Frobenius norm as $\|\mathcal{X}\|_F = \sqrt{\sum_{i_1 i_2 \cdots i_n} (x_{i_1 i_2 \cdots i_n})^2}$ and the nuclear norm of matrix as $\|X\|_* = \sum_i \sigma_i$, where σ_i is the i-th largest singular value of matrix X. For a vector $\boldsymbol{\alpha}$, the l_2-norm is $\|\boldsymbol{\alpha}\|_2 = \sqrt{\sum_i \alpha_i^2}$. For a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, we use $\hat{\mathcal{X}}$ to denote the result of discrete Fourier transformation of $\mathcal{X}$ along the 3-rd dimension, i.e., $\hat{\mathcal{X}} = \text{fft}(\mathcal{X}, [], 3)$. Contrarily, we can compute $\mathcal{X}$ from $\hat{\mathcal{X}}$ using the inverse FFT, i.e., $\mathcal{X} = \text{ifft}(\hat{\mathcal{X}}, [], 3)$.

The work[35] give the first definition of the **block circulation** operation for a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$:

$$\text{bcirc}(\mathcal{X}) := \begin{bmatrix} X^{(1)} & X^{(n_3)} & \cdots & X^{(2)} \\ X^{(2)} & X^{(1)} & \cdots & X^{(3)} \\ \vdots & \vdots & \ddots & \vdots \\ X^{(n_3)} & X^{(n_3-1)} & \cdots & X^{(1)} \end{bmatrix} \in \mathbb{R}^{n_1 n_3 \times n_2 n_3}.$$

The **block diagonalization** matrix of $\mathcal{X}$ is defined as

$$\text{bdiag}(\mathcal{X}) := \begin{bmatrix} X^{(1)} & & & \\ & X^{(2)} & & \\ & & \ddots & \\ & & & X^{(n_3)} \end{bmatrix} \in \mathbb{R}^{n_1 n_3 \times n_2 n_3}.$$

The block circulant matrix can be block diagonalized, i.e.,

$$\text{bdiag}(\hat{\mathcal{X}}) = (F_{n_3} \otimes I_{n_1}) \cdot \text{bcirc}(\mathcal{X}) \cdot (F_{n_3}^H \otimes I_{n_2}),$$

where $F_{n_3} \in \mathbb{C}^{n_3 \times n_3}$ is the discrete Fourier transformation matrix, $I_n \in \mathbb{R}^{n \times n}$ is an identity matrix, $\otimes$ denotes the Kronecker product. We also define the following operator

$$\text{bvec}(\mathcal{X}) = \begin{bmatrix} X^{(1)} \\ X^{(2)} \\ \vdots \\ X^{(n_3)} \end{bmatrix}, \text{bvfold}(\text{bvec}(\mathcal{X})) = \mathcal{X}.$$

2.2. Tensor preliminaries

Definition 1 (Tucker Decomposition and Tucker Rank)**.** [5] The Tucker decomposition is a form of higher-order PCA. It decomposes the tensor into a core tensor multiplied by a matrix along each mode. Given a tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, it can be decomposed as:

$$\mathcal{X} \approx \mathcal{G} \times_1 A \times_2 B \times_3 C = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} a_p \circ b_q \circ c_r,$$

where "$\times_n$" denotes mode-n product, "$\circ$" denotes the outer product, $A \in \mathbb{R}^{I \times P}$, $B \in \mathbb{R}^{J \times Q}$, $C \in \mathbb{R}^{K \times R}$ are factor matrices and $\mathcal{G} \in \mathbb{R}^{P \times Q \times R}$ is the core tensor.

A Tucker rank (also known as n-rank) of an N-order tensor is defined the rank of unfolding matrix along each mode, $\boldsymbol{r} = (r_1, r_2, \ldots, r_N)$, where r_n, $n = 1, 2, \ldots, N$, denotes the rank of $X_{(n)}$.

Definition 2 (Sum of Nuclear Norm (SNN))**.** [4] For an N-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}$, the definition of tensor SNN is the weighted average of the nuclear norm of all matrices unfolded along each mode. Mathematically, it can be formulated as:

$$\|\mathcal{X}\|_{*_s} := \sum_{i=1}^{N} \alpha_i \left\| X_{(i)} \right\|_*,$$

where α_i denotes the weight of matrix unfolded along i-th mode.

Definition 3 (t-product)**.** [35] The t-product between two 3-order tensors $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $\mathcal{Y} \in \mathbb{R}^{n_2 \times n_4 \times n_3}$ is defined as:

$$\mathcal{Z} = \mathcal{X} * \mathcal{Y} := \text{bvfold}(\text{bcirc}(\mathcal{X})\text{bvec}(\mathcal{Y})).$$

Using the above property, the t-product can be written as:

$$\hat{\mathcal{Z}} = \text{bvfold}(\text{bdiag}(\hat{\mathcal{X}})\text{bvec}(\hat{\mathcal{Y}})),$$

Definition 4 (Other Special Tensor)**.** [35] The **identity tensor** $\mathcal{I} \in \mathbb{R}^{n \times n \times n_3}$ is the tensor whose first frontal slice is the $n \times n$ identity matrix, and other frontal slices are all zeros. The **orthogonal tensor** $\mathcal{Q} \in \mathbb{R}^{n \times n \times n_3}$ is satisfies $\mathcal{Q} * \mathcal{Q}^T = \mathcal{Q}^T * \mathcal{Q} = \mathcal{I}$. A tensor $\mathcal{X}$ is called **f-diagonal** if each frontal slice $\mathcal{X}^{(i)}$ is a diagonal matrix.

Theorem 1 (t-SVD)**.** *[35] Let $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$. Then it can be factored as*

$$\mathcal{X} = \mathcal{U} * \mathcal{S} * \mathcal{V}^H,$$

where $\mathcal{U} \in \mathbb{R}^{n_1 \times n_1 \times n_3}$, $\mathcal{V} \in \mathbb{R}^{n_2 \times n_2 \times n_3}$ are orthogonal, and $\mathcal{S} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ is an f-diagonal tensor. The t-SVD based on t-product, which can be efficiently obtained by computing a series of matrix SVDs in the Fourier domain.

Definition 5 (Tensor Tubal Rank)**.** [11] The tensor tubal rank denoted as $rank_t(\mathcal{X})$, is defined as the number of non-zero tubes of $\mathcal{S}$, where $\mathcal{S}$ is from the t-SVD of $\mathcal{X} = \mathcal{U} * \mathcal{S} * \mathcal{V}^H$, that is

$$\text{rank}_t(A) = \#\{i : \mathcal{S}(i, :, :) \neq 0\}.$$

Definition 6 (Tensor Nuclear Norm (TNN))**.** [11] Let $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, the definition of tensor TNN is the sum of singular values of all the frontal slices of $\hat{\mathcal{X}}$, it can be formulated as:

$$\|\mathcal{X}\|_{*_t} := \sum_{i=1}^{n_3} \|\hat{X}^{(i)}\|_*.$$

2.3. Convolutional sparse coding

Unlike the traditional sparse coding model, CSC replaces the general linear representation with a sum of convolutions between a set of filters $\boldsymbol{d_i}$ (the set also known as a dictionary) and its corresponding feature maps $\boldsymbol{\Gamma_i}$ [26]. Besides, the patch-based dictionary restores the same information in the image several times, while CSC integrates these information only once. One assumes that an signal $X \in \mathbb{R}^N$ admits a decomposition as $X = \sum_{i=1}^{m} \boldsymbol{d_i} * \boldsymbol{\Gamma_i}$, where $\boldsymbol{d_i} \in \mathbb{R}^n$ denotes the filters, $\boldsymbol{\Gamma_i} \in \mathbb{R}^N$ is the feature map corresponding to $\boldsymbol{d_i}$, Λ is the trade-off parameter and $*$ is the convolution

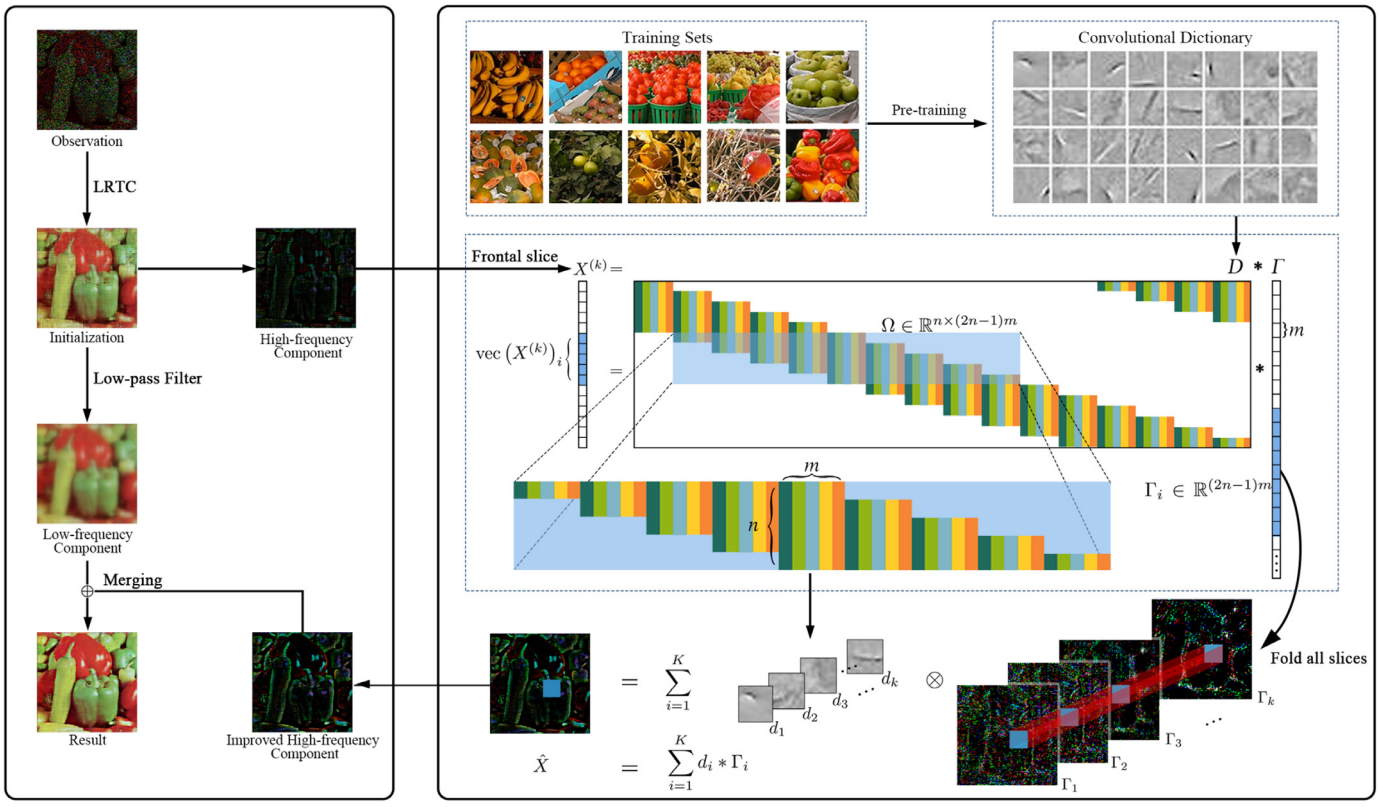


Fig. 1. Flowchart of LRTC-CSC models for low-rank tensor completion. The first step is to generate an initial tensor according to observations. Then a low-pass filter is utilized to divide the initial tensor into two components, a high-frequency and a low-frequency one. Next, the high-frequency component is processed by CSC model, in which the dictionary D (shown in the format of convolutional matrix) is pre-trained by a small-sample dataset. The stripe dictionary Ω, which is of size $n \times (2n-1)m$, is obtained by extracting the i-th patch from the global convolutional dictionary D. The stripe vector Γ_i, which is the corresponding sparse respresent, contains all coefficients of atoms contributing to $vec(X^{(k)})_i$. At last, by merging the improved high-frequency component and the original low-frequency one, the final result is reconstructed.

operator. The most univesal form of CSC is Convolutional Basis Pursuit DeNoising (CBPDN), formulated as:

$$\arg\min_{\Gamma_i} \quad \frac{1}{2}\left\|\sum_{i=1}^{m} \boldsymbol{d_i} * \boldsymbol{\Gamma_i} - X\right\|_2^2 + \Lambda \sum_{i=1}^{m} \|\boldsymbol{\Gamma_i}\|_1. \tag{4}$$

In contrast to linear representation, convolutional dictionary can acquire shift-invariant features, which improves the efficiency of the model.

To facilitate understanding, we show the process of CSC calculation in Fig. 1. Following Fig. 1 the above can be written in matrix form as $X = \sum_{i=1}^{m} \boldsymbol{d_i} * \boldsymbol{\Gamma_i} = D\Gamma$, where $D \in \mathbb{R}^{N \times Nm}$ is a banded convolutional dictionary, $\Gamma \in \mathbb{R}^{Nm}$ is a global sparse representation. Define $R_i \in \mathbb{R}^{n \times N}$ as the operator that extracts the i-th n-dimensional patch from X, from in the graph, we denote by $(\cdot)_i$, i.e., $R_i X = X_i$. Therefore a patch X obtained from the global signal is equal to $\Omega\Gamma_i$, where $\Omega \in \mathbb{R}^{n \times (2n-1)m}$ is a strip dictionary and $\Gamma_i \in \mathbb{R}^{(2n-1)m}$ is a strip vector.

3. The proposed model and algorithm

It's imperfect for LRTC models that they can only recover the low-rank component of underlying tensors. In the meanwhile, lots of details are ignored, which derives many composite algorithms with extra priors emerged to improve the missing details. These details often represent the local features or high-frequency component of an image. Previous studies about TV regularization have shown its over-patch (over-smooth) effects. To obtain better details when recovering underlying tensors, CSC is introduced to improve the high-frequency component of the underlying data. It's known that the high-frequency component often contains specific detailed information against the low-frequency one. Thus, it's a feasible choice to improve the high-frequency component solely. By using a pre-trained convolutional dictionary, the corresponding feature maps can be iteratively calculated to approximate the incomplete high-frequency component of tensor data, which is produced by LRTC models. Due to the overcompleteness of the dictionary, the convolution results of obtained feature maps and dictionary would be an improved version of the previous high-frequency component. In this section, we propose to incorporate CSC for different LRTC methods to verify the effectiveness of CSC regularization. Since the low-rank prior of LRTC can be approximated by SNN and TNN, two LRTC-CSC compound methods are developed.

Method LRTC-CSC-I

As the tratidional typical low-rank approximation method, SNN has gained unexpected achievements in LRTC problems. Even though, its promise is limited by the tensor mode-n flattening operator, which breaks the global structure of the whole tensor. To improve the performance of SNN-based model, we consider to impose a CSC regularization onto the detail component of underlying tensor, mathematically formulated as:

$$\arg\min_{\mathcal{X}} \quad \sum_{k=1}^{N} \alpha_k \left\|X_{(k)}\right\|_{*_s} + \lambda\Phi(\mathcal{X}) \quad \text{s.t.} \quad \mathcal{X}_\Omega = \mathcal{T}_\Omega, \tag{5}$$

where $\Phi(\mathcal{X})$ denotes the regularization item, i.e. CSC, and λ is a trade-off parameter.

The constraint condition $\mathcal{X}_\Omega = \mathcal{T}_\Omega$ can be rewritten as a function like:

$$\min \quad P(\mathcal{X}) = \begin{cases} 0, & \text{if} \quad \mathcal{X}_\Omega = \mathcal{T}_\Omega, \\ \infty, & \text{otherwise.} \end{cases} \tag{6}$$

By introducing auxiliary variables $\{F_k\}_{k=1}^N$ to disentangle the relationship between $\{X_{(k)}\}_{k=1}^N$ and $\mathcal{Z} = \mathcal{X}$, the augmented Lagrangian function of (5) can be formulated as:

$$\begin{aligned} \mathcal{L} = & \sum_{k=1}^{N} (\alpha_k \|F_k\|_{*_s} + \tfrac{\beta_1}{2} \left\| F_k - X_{(k)} \right\|_F^2 + \left\langle W_{1_k}, F_k - X_{(k)} \right\rangle) \\ & + \tfrac{\beta_2}{2} \|\mathcal{Z} - \mathcal{X}\|_F^2 + \langle \mathcal{W}_2, \mathcal{Z} - \mathcal{X} \rangle + P(\mathcal{X}) + \lambda \Phi(\mathcal{Z}), \end{aligned} \tag{7}$$

where the matrix $\{W_{1_k}\}_{k=1}^N$ and the tensor $\mathcal{W}_2$ are Lagrangian multipliers and β_1 and β_2 are the penalty parameter.

Under the iterative optimization framework of ADMM [36], the solution process of (7) can be concluded by solving three subproblems, i.e. $[F, \mathcal{Z}]$,$\mathcal{X}$, W_1 and $\mathcal{W}_2$.

3.1. Solving $[F, \mathcal{Z}]$-subproblem

In this part, we are going to solve the $[F, \mathcal{Z}]$-subproblem separately, since they are decoupled.

1) By fixing $\mathcal{X}$ and $\mathcal{Z}$, rewrite the function of F-subproblem:

$$\underset{F_k}{argmin} \quad \sum_{k=1}^{N} \left(\alpha_k \|F_k\|_* + \frac{\beta_1}{2} \left\| F_k - X_{(k)} + \frac{W_{1_k}}{\beta_1} \right\|_F^2 \right). \tag{8}$$

For each auxiliary variable F_k in $\{F_k\}_{k=1}^N$, the iterative closed-form solution of (8) is:

$$F_k^{i+1} = \mathbf{D}_{\tau_k}(X_{(k)}^i - \frac{W_{1_k}^i}{\beta_1}) = U \Sigma_{\tau_k} V^T, \tag{9}$$

where $\tau_k = \frac{\alpha_k}{\beta_1}$, and i is the number of updates. The operator $D_\tau(X)$ denotes the thresholding singular value decomposition (SVD), which is defined as $D_\tau(X) = U \Sigma_\tau V^T$ [4], where $\Sigma_\tau = \text{diag}(\max(\sigma_i - \tau, 0))$, the σ_i is the i-th largest singular value of matrix X, and diag$(\cdot)$ denotes a diagnal matrix.

2) By fixing $\mathcal{X}$ and F, the $\mathcal{Z}$-subproblem is:

$$\arg\min_{\mathcal{Z}} \quad \tfrac{\beta_2}{2} \left\| \mathcal{Z} - (\mathcal{X} - \tfrac{W_2}{\beta_2}) \right\|_F^2 + \lambda \Phi(\mathcal{Z}). \tag{10}$$

Actually as the plug-and-play framework suggested in [32], the problem (10) can be regarded as a new denoising problem. $\Phi(\mathcal{Z})$ measures the degree of noise in $\mathcal{Z}$, the smaller, the better. By treating " $\mathcal{X} - \frac{\omega_2}{\beta_2}$ " as a noisy image and "$\mathcal{Z}$" as a clean image, CSC model becomes the denoiser to solve $\mathcal{Z}$-subproblem and outputs a clean image for comparison. Thus, we use the noise image " $\mathcal{X} - \frac{\omega_2}{\beta_2}$ " to be the input of CSC model, which can be explicitly formulated as:

$$\arg\min_{M_i} \quad \tfrac{1}{2} \left\| \sum_{i=1}^{K} d_i * M_i - \mathcal{U} \right\|_F^2 + \Lambda \sum_{i=1}^{K} \|M_i\|_1, \tag{11}$$

where $\mathcal{U} = \mathcal{X} - \frac{\omega_2}{\beta_2}$, Λ is the hyper-parameter, and K is the number of filters. Note that $\mathcal{U}$ is a 3-rd tensor while d_i, M_i are matrices. To keep consistency, $\mathcal{U}$ is cut into three frontal slices for three different color channels in color images, repectively. And the following operations are completed at the level of matrices.

The dictionary learning algorithm is based on ADMM consensus dictionary update [26], which learns the dictionary from the high-frequency components of the input image. Thus, we obtain the component after subtracting a low-frequency computed by Tikhonov regularization [37] from the image. That is, only the high-frequency component of $\mathcal{U}$ is processed by CSC [29], thus, we use a low-pass filter to acquire the high-frequency component of $\mathcal{U}$ and in the rest of this section, $\mathcal{U}$ represents its high-frequency component. At last, the underlying result can be obtained by summing the low-frequency component and recovered high-frequency component.

Because the inaccurate filters will lead to producing new artifacts or structure loss problems, an extra gradient constraint is adopted to suppress the outliers based on the original CSC model. What's more, as suggested in [26], the gradient constraint on the feature maps is superior to applying that on the image domain. Thus, we further utilize problem (11) with gradient constraint on the feature maps, and it can be explicitly formulated as:

$$\begin{aligned} \arg\min_{M_i} \quad & \tfrac{1}{2} \left\| \sum_{i=1}^{K} d_i * M_i - \mathcal{U} \right\|_F^2 + \Lambda \sum_{i=1}^{K} \|M_i\|_1 \\ & + \tfrac{\tau}{2} \sum_{i=1}^{K} (\|g_0 * M_i\|_F^2 + \|g_1 * M_i\|_F^2), \end{aligned} \tag{12}$$

where g_0 and g_1 are the filters that compute the gradients along image rows and columns respectively, and τ is the hyperparameter. By introducing linear operators $G_j M_i = g_j * M_i$, the gradient constraint term of (12) can be rewritten as:

$$\frac{\tau}{2} \sum_{i=1}^{K} \left(\|G_0 M_i\|_F^2 + \|G_1 M_i\|_F^2 \right). \tag{13}$$

Further more, considering the conception of block matrix, (13) can be simplified as:

$$\frac{\tau}{2} \|\Psi_0 \mathcal{M}\|_F^2 + \frac{\tau}{2} \|\Psi_1 \mathcal{M}\|_F^2, \tag{14}$$

where

$$\Psi_j = \begin{pmatrix} G_j & 0 & \cdots \\ 0 & G_j & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}, \tag{15}$$

and $\mathcal{M} = (M_1 M_2 \cdots M_K)^T$.

Let $D_i M_i = d_i * M_i$, $\mathcal{D} = (D_1 D_2 \cdots D_K)$, the augmented Lagrangian function of (12) is:

$$\begin{aligned} L_2 = & \tfrac{1}{2} \|\mathcal{D}\mathcal{M} - \mathcal{U}\|_F^2 + \Lambda \|\mathcal{B}\|_1 + \langle \Theta, \mathcal{M} - \mathcal{B} \rangle \\ & + \tfrac{\rho}{2} \|\mathcal{M} - \mathcal{B}\|_F^2 + \tfrac{\tau}{2} \|\Psi_0 \mathcal{M}\|_F^2 + \tfrac{\tau}{2} \|\Psi_1 \mathcal{M}\|_F^2, \end{aligned} \tag{16}$$

where $\mathcal{B}$ is introduced as an auxiliary variables satisfying $\mathcal{B} = \mathcal{M}$, Θ is the Lagrangian multiplier and ρ is the penalty parameter.

According to the inexact ADMM framework [34], the problem (16) can be solved by solving a sequence of subproblems.

1) Rewrite the $\mathcal{M}$-subproblem:

$$\begin{aligned} \arg\min_{\mathcal{M}} \quad & \tfrac{1}{2} \|\mathcal{D}\mathcal{M} - \mathcal{U}\|_F^2 + \tfrac{\rho}{2} \|\mathcal{M} - \mathcal{B} + C\|_F^2 \\ & + \tfrac{\tau}{2} \|\Psi_0 \mathcal{M}\|_F^2 + \tfrac{\tau}{2} \|\Psi_1 \mathcal{M}\|_F^2, \end{aligned} \tag{17}$$

where C denotes the parameter item $C = \frac{\Theta}{\rho}$. Considering that the derivative of the convolution operation is difficult to obtain in space domain, we transform (17) to the Fourier domain and it can be formulated as:

$$\begin{aligned} \arg\min_{\hat{\mathcal{M}}} \quad & \tfrac{1}{2} \left\| \hat{\mathcal{D}}\hat{\mathcal{M}} - \hat{\mathcal{U}} \right\|_F^2 + \tfrac{\rho}{2} \left\| \hat{\mathcal{M}} - \hat{\mathcal{B}} + \hat{\mathcal{C}} \right\|_F^2 \\ & + \tfrac{\tau}{2} \left\| \hat{\Psi}_0 \hat{\mathcal{M}} \right\|_F^2 + \tfrac{\tau}{2} \left\| \hat{\Psi}_1 \hat{\mathcal{M}} \right\|_F^2, \end{aligned} \tag{18}$$

where $\hat{\mathcal{D}}$, $\hat{\mathcal{M}}$, $\hat{\mathcal{U}}$, $\hat{\mathcal{B}}$, $\hat{\mathcal{C}}$, $\hat{\Psi}_0$ and $\hat{\Psi}_1$ denotes the corresponding expressions in the Fourier domain. A closed-form solution of (18) is:

$$\left(\hat{\mathcal{D}}^H \hat{\mathcal{D}} + \rho I + \tau \hat{\Psi}_0^H \hat{\Psi}_0 + \tau \hat{\Psi}_1^H \hat{\Psi}_1 \right) \hat{\mathcal{M}}^{j+1} = \hat{\mathcal{D}}^H \hat{\mathcal{U}} + \rho \left(\hat{\mathcal{B}}^j - \hat{\mathcal{C}}^j \right), \tag{19}$$

which can be solved by iterated application of the Sherman-Morrison formula [38]. The superscript j indicates the number of

updates. After obtaining $\hat{\mathcal{M}}$, the corresponding $\mathcal{M}$ in spatial domain can be calcaulated by:

$$\mathcal{M}^{j+1} = \text{ifft}(\hat{\mathcal{M}}^{j+1}), \tag{20}$$

where ifft$(\cdot)$ denotes the inverse Fourier transform.

2) Rewrite the $\mathcal{B}$-subproblem:

$$\arg\min_{\mathcal{B}} \quad \Lambda\|\mathcal{B}\|_1 + \tfrac{\rho}{2}\|\mathcal{M} - \mathcal{B} + C\|_F^2. \tag{21}$$

By using soft-thresholding algorithm, the closed-form solution of (21) can be obtained as:

$$\mathcal{B}^{j+1} = \mathrm{S}_{\Lambda/\rho}(\mathcal{M}^{j+1} + C^j). \tag{22}$$

where $S_l(\cdot)$ denotes the soft-thresholding function, which can be calculated as: $S_l(\boldsymbol{x}) = sign(x)max(0, |\boldsymbol{x}| - l)$.

3) Update C by fixing $\mathcal{M}$ and $\mathcal{B}$:

$$C^{j+1} = C^j + \Lambda\left(\mathcal{M}^{j+1} - \mathcal{B}^{j+1}\right). \tag{23}$$

At last, the outputing high-frequency component of $\mathcal{Z}$ can be calculated by:

$$\mathcal{Z}^{i+1} = \mathcal{D}\mathcal{M}^*, \tag{24}$$

where $\mathcal{M}^*$ is the optimal solution of $\mathcal{M}$. And after adding the low-frequency component of $\mathcal{U}$, an entire solution $\mathcal{Z}$ is obtained.

3.2. Solving $\mathcal{X}$-subproblem

By fixing F and $\mathcal{Z}$, the $\mathcal{X}$-subproblem can be simplified and rewritten as:

$$\begin{aligned}\arg\min_{\mathcal{X}} \quad & \sum_{k=1}^{N} \tfrac{\beta_1}{2}\left\|F_k - X_{(k)} + \frac{W_{1_k}}{\beta_1}\right\|_F^2 + P(\mathcal{X}) \\ & + \tfrac{\beta_2}{2}\left\|\mathcal{Z} - \mathcal{X} + \frac{\mathcal{W}_2}{\beta_2}\right\|_F^2.\end{aligned} \tag{25}$$

The closed-form solution of (25) can be obtained by:

$$\mathcal{X}^{i+1} = \left(\frac{\beta_1}{\beta_1+\beta_2}\frac{\sum_{k=1}^{N} fold\left(F_k^{i+1} + \frac{W_{1_k}^i}{\beta_1}\right)}{N} + \frac{\beta_2}{\beta_1+\beta_2}\left(\mathcal{Z}^{i+1} + \frac{\mathcal{W}_2^i}{\beta_2}\right)\right)_{\bar{\Omega}} + \mathcal{T} \tag{26}$$

where $\bar{\Omega}$ is the complement set of Ω.

3.3. Solving $[W_{1_k}, \mathcal{W}_2]$-subproblem

By fixing $\mathcal{X}$ and F, $\mathcal{X}$ and $\mathcal{Z}$, we can update the Lagrangian multiplier W_1^{i+1} and $\mathcal{W}_2^{i+1}$, respectively.

For each value W_{1_k} in the multiplier $\{W_{1_k}\}_{k=1}^N$:

$$W_{1_k}^{i+1} = W_{1_k}^i + \beta_1\left(F_k^{i+1} - X_{(k)}^{i+1}\right), \tag{27}$$

and

$$\mathcal{W}_2^{i+1} = \mathcal{W}_2^i + \beta_2\left(\mathcal{Z}^{i+1} - \mathcal{X}^{i+1}\right). \tag{28}$$

The overall pseudocode of LRTC-CSC-I algorithm is summarized in Algorithm 1. Actually, our model can degenerate to LRTC-SNN model [4] by setting the hyperparameter $\lambda = 0$.

Method LRTC-CSC-II

Similar to method I, we use the TNN to approximate the low-rank prior of tensor, instead. The LRTC-CSC-II model is mathematically formulated as:

$$\arg\min_{\mathcal{X}} \quad \sum_{k=1}^{n_3}\|\hat{X}^{(k)}\|_{*_t} + \lambda\Phi(\mathcal{X}) \quad \text{s.t.} \quad \mathcal{X}_\Omega = \mathcal{T}_\Omega. \tag{29}$$

Algorithm 1 LRTC-CSC-I algorithm.

Input: index set Ω, original data $\mathcal{T}$, filters set $\mathcal{D}$
Parameters: $\Lambda, \lambda, \rho, \tau, \alpha_k, W_{1_k}, \mathcal{W}_2, \beta_1, \beta_2, \epsilon$
Initialization: $\mathcal{X}_\Omega^0 = \mathcal{T}_\Omega$, $\mathcal{X}_{\bar{\Omega}}^0 = \text{mean}(\mathcal{T})$
repeat
  F-subproblem
  Update F_k^{i+1} by (9)
  $\mathcal{Z}$**-subproblem**
  for $j = 1$ to MaxIter **do**
    Update $\mathcal{M}^{j+1}$ by (20)
    Update $\mathcal{B}^{j+1}$ by (22)
    Update C^{j+1} by (23)
  end for
  Update $\mathcal{Z}^{i+1}$ by (24)
  $\mathcal{X}$**-subproblem**
  Update $\mathcal{X}^{i+1}$ by (26)
  W_{1_k}, $\mathcal{W}_2$**-subproblem**
  Update $W_{1_k}^{i+1}$ by (27)
  Update $\mathcal{M}_2^{i+1}$ by (28)
until Reach convergence $\|\mathcal{X} - \mathcal{T}\|_F / \|\mathcal{T}\|_F < \epsilon$
Output:The recovered tensor $\mathcal{X}$

The constraint condition can be rewritten as (6). By introducing auxiliary variables $\mathcal{F} = \mathcal{X}$ and $\mathcal{Z} = \mathcal{X}$, the augmented Lagrangian function of (29) can be formulated as:

$$\begin{aligned}\mathcal{L} = & \sum_{k=1}^{n_3}\left(\|\hat{F}^{(k)}\|_{*_t} + \tfrac{\beta_1}{2}\|F^{(k)} - X^{(k)}\|_F^2 + \langle W_1^{(k)}, F^{(k)} - X^{(k)}\rangle\right) \\ & + \tfrac{\beta_2}{2}\|\mathcal{Z} - \mathcal{X}\|_F^2 + \langle\mathcal{W}_2, \mathcal{Z} - \mathcal{X}\rangle + P(\mathcal{X}) + \lambda\Phi(\mathcal{Z}).\end{aligned} \tag{30}$$

where $X^{(k)}$ is used to represent $\mathcal{X}(:,:,k)$ and the tensor $\mathcal{W}_1$ and $\mathcal{W}_2$ are Lagrangian multipliers and β_1 and β_2 are the penalty parameter.

Note that Eq. (30) is different from Eq. (7). For a 3-order tensor $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times I_3}$. The SNN is done for each mode of the tensor, i.e.,$\sum_{k=1}^N \|X_{(k)}\|_*$ here $N = 3$, and $X_{(k)} \in \mathbb{R}^{I_k \times \prod_{i\neq k}^3 I_i}$. The TNN is done for each frontal slice of the tensor, i.e., $\sum_{k=1}^{I_3}\|\hat{X}^{(i)}\|_*$ here $X^{(k)} \in \mathbb{R}^{I_1\times I_2}$.

In order to distinguish the k-th frontal slice and the k-th iteration times of a tensor, we use $\mathcal{X}(:,:,k)$ to denote the k-th frontal slice and $\mathcal{X}^k$ for the k-th iteration. We divide the problem (30) into three subproblems, i.e. $[\mathcal{F}, \mathcal{Z}]$, $\mathcal{X}$, $\mathcal{W}_1$ and $\mathcal{W}_2$, and solve the problem with the same optimization framework of ADMM. For the sake of brevity, we describe how to solve the LRTC-CSC-II model in Appendix A. Actually, our model can degenerate to LRTC-TNN model [11] by setting the hyperparameter $\lambda = 0$.

4. Numerical experiments

In this section, we verified our LRTC-CSC models with popular color image datasets, such as *Peppers, Lena, Starfish* and so on. Besides, MRI data and video are also leveraged to our benchmark, which shows the generalization of LRTC-CSC model. Meanwhile, several state-of-the-art LRTC methods are compared in different situations:

1. LRTC-SNN(SNN) [4]: It adopted SNN to approximate the low-rank prior of underlying tensors without any regularizations in the objective function.
2. LRTC-TNN(TNN) [11]: It used TNN to describe the low-rank prior of underlying tensors without any regularizations in the objective function.
3. LRTC-TV(TV) [13]: It combined the Tucker rank-based SNN fidelity item and TV regularization to approximate the global low-rank structure and local smoothness, respectively.

Fig. 2. The 10 fruit images for training.

4. TNN-3DTV(3DTV) [16]: It combined TNN and anisotopic TV regularization, inorder to extract the intrinsic structures of visual data and exploit the local smooth and piecewise priors, simultaneously.
5. MF-Framelet(Fram) [14]: It leveraged matrix factorization-based SNN to capture the global structure of underlying tensor with the Framelet regularization.
6. Framelet-TV(FTV) [39]: It combined the TT rank-based SNN and the hybrid regularization of Framelet and TV, aiming at characterizing the global TT low-rankness, capturing the abundant details and enhancing the temporal smoothness of the tensor, respectively.
7. TNN-DCT(DCT) [40]: It is inspired by the invertible linear transforms based tensor-tensor product, which extends the traditional TNN to a more general format. Different from LRTC-TNN, the Discrete Fourier Transform is replaced by Discrete Cosine Transform.
8. WTNNDL(DL) [41]: It employed a weighted TNN of the tensor to approximate the global structure of the data and uses sparse coding to elucidate the local patterns of the data.
9. LRSSRTC(SSR) [42]: It integrated SNN-based low-rank tensor completion and sparse self-representation into a unified framework and proposes a new completion model.

To measure the recovering performance of various models, both the Peak Signal to Noise Ratio (PSNR) and Structural SIMilarity Index (SSIM) are used. In consideration of the multiband structure of tensor data, we adopt the mean value of all bands as the evaluation metric.

All experiments are performed in MATLAB R2020a in Linux with an Inter(R) Core(TM) i7-9800X CPU at 3.80 GHz and 64GB RAM. To accelerate the running speed, we performed the CSC phase with a graphics processing unit (GPU) GeForce RTX 2080 Ti.

4.1. Pre-training CSC

The convolutional filters are pre-trained with the popular fruit dataset [24] in color image experiments as shown in Fig. 2. To make sure the optimal number of filters, we trained several dictionaries of different number of filters with a fruit dataset. And then these dictionaries were used to our benchmark for the sampling rate 20%. The experimental results are shown in Fig. 3. The best performance corresponds to the best number of filters. All the results in the left figure of Fig. 3 indicated that the proposed model would gain the best performance when a dictionary consists of 32 filters. Except for the *Lena* image, the other results in the right figure of Fig. 3 also show the superiority when the filter number is 32. Thus, we trained a dictionary of 32 filters with the fruit dataset beforehand. These filters are of size 16×16, $\Lambda = 10$, and $\tau = 0.1$. It's worth noting that the computation of convolution is finished in the Fourier domain, which can be regarded as dot product, thus the size of filters can be even.

The penalty factor Λ in Eq. (4) determines how clean the learned dictionary is during the pre-traing process. To investigate the impact of Λ, we select Λ from $\{0.001, 0.01, 0.1, 1, 10, 100\}$ to conduct experiments on the fruit dataset as shown in Fig. 2. The visual results of the learned dictionaries can be found in Fig. 4. What's more, Fig. 5(a) shows the convergence behavior of CSC model on different Λs and it's obvious that almost all the models converge to a optimal point in a normal way except for $\Lambda = 0.001$, which indicates the overfitting phenomenon. Fig. 5 (b) shows the performance of utilizing these dictionaries on color images recovery experiments. When $\Lambda = 10$, the model can achieve a highest PSNR value.

4.2. Color images

Before the experiments begin, we execute the missing process to original images. Specifically, to obtain observed data, all the elements in color images will be erased randomly according to a setting ratio (same in the following MRI and video experiments).

Fig. 6 shows the recovered results of color images by all comparing models. The first six images are $256 \times 256 \times 3$ in size, while the last two images are $256 \times 192 \times 3$ in size. The details are marked out and enlarged in the mark boxes. Comparing to the degradation models of the proposed ones, LRTC-SNN and LRTC-TNN, it's intuitionistic to show the effect of CSC prior. The Framelet-TV method is very competitive to our model LRTC-CSC-I, however, the proposed model holds more clean and smooth details even in an inferior comparison on last two color images. In most situations, LRTC-CSC-I can obtain a higher performance. And the results of LRTC-CSC-II are the most similar to the original images, especially.

The PSNR and SSIM values of recovered results for the 8 color images by different algorithms are shown in Tab.1 and 2. One can see that the LRTC-TNN method is superior to SNN-based LRTC-SNN on the whole. LRTC-TV is very closed to TNN-3DTV on SSIM values but inferior on PSNR values. The DCT method is the improved version of TNN without other regularizations indeed, thus can hardly compete with other composite models. And Framelet-TV method holds onto the position of the third place by a comfortable margin in most situations. Its performance ends up second only to the proposed two models. Neither the WTNNDL method nor the LRSSRTC method performs well on this dataset. WTNNDL uses its own observations for dictionary learning, while our dictionary is learned by an external prior, thus we learn wealthier features. Especially, the LRTC-CSC models have a larger lead at color images similar to peppers and fruits due to the dictionary trained by a fruit dataset, which also shows the importance of the relationship between underlying data and dataset. The results of Framelet-TV on the last two images are superior to our CSC-I method, which shows the shortcomings of SNN-based methods. The structure features are broken by unfolding tensors into matrices along each mode. What's more, the few relationships between underlying tensors and dataset exacerbate the issue. Besides, when the missing ratio is 90%, other algorithms can hardly hold a stable SSIM value (especially the last three images), LRTC-CSC models are still standing at a relatively high level.

4.3. MRI dataset

In this subsection, an MRI dataset of size $180 \times 216 \times 30$ is chosen to test our model. Except for the 30 bands used for the test, we select another 10 bands from the data as the dictionary training set. And different from the color image experiments, the

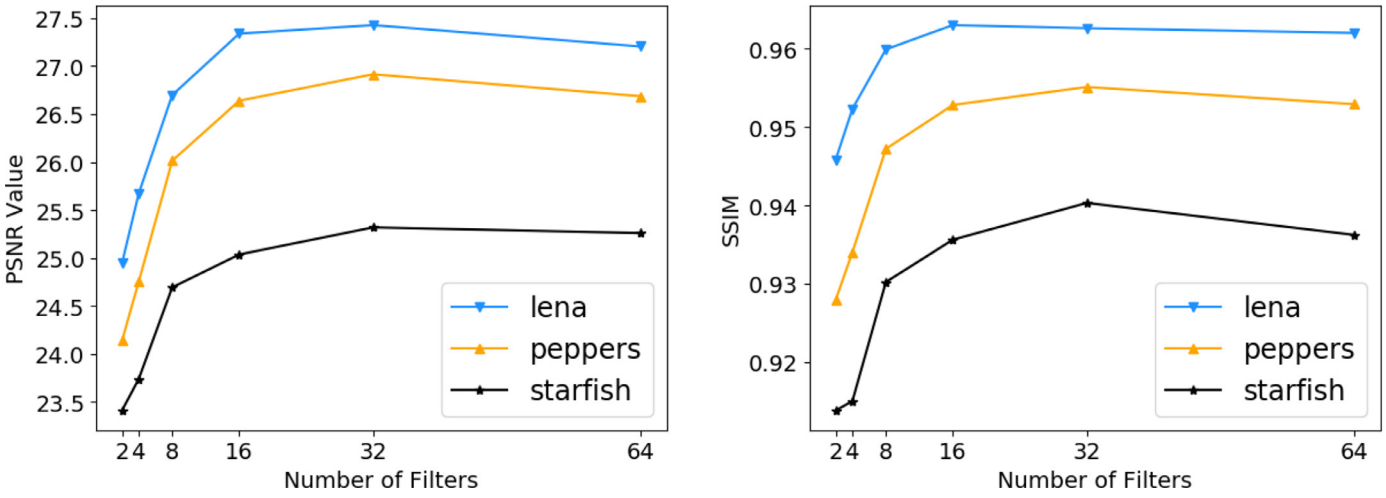


Fig. 3. The PSNR and SSIM values of color images recovered by LRTC-CSC-I when different number of filters are used in a dictionary for the sampling rate 20%.

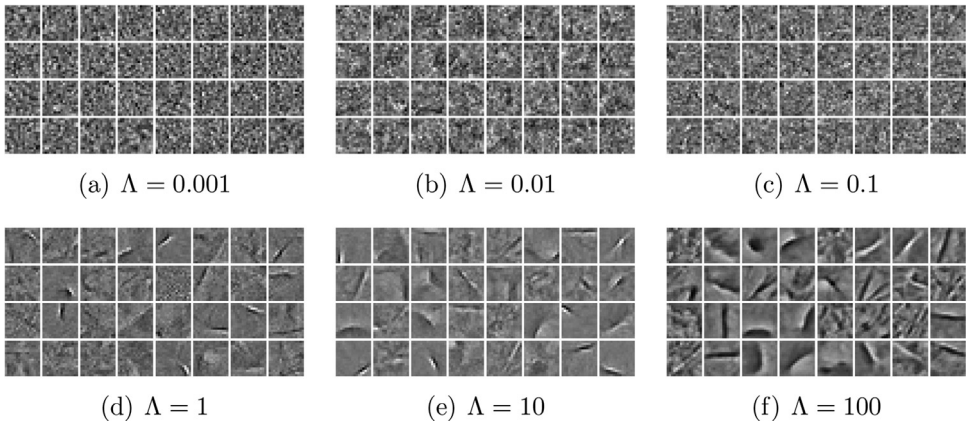


Fig. 4. The dictionary of 32 filters trained with fruit dataset on different Λs.

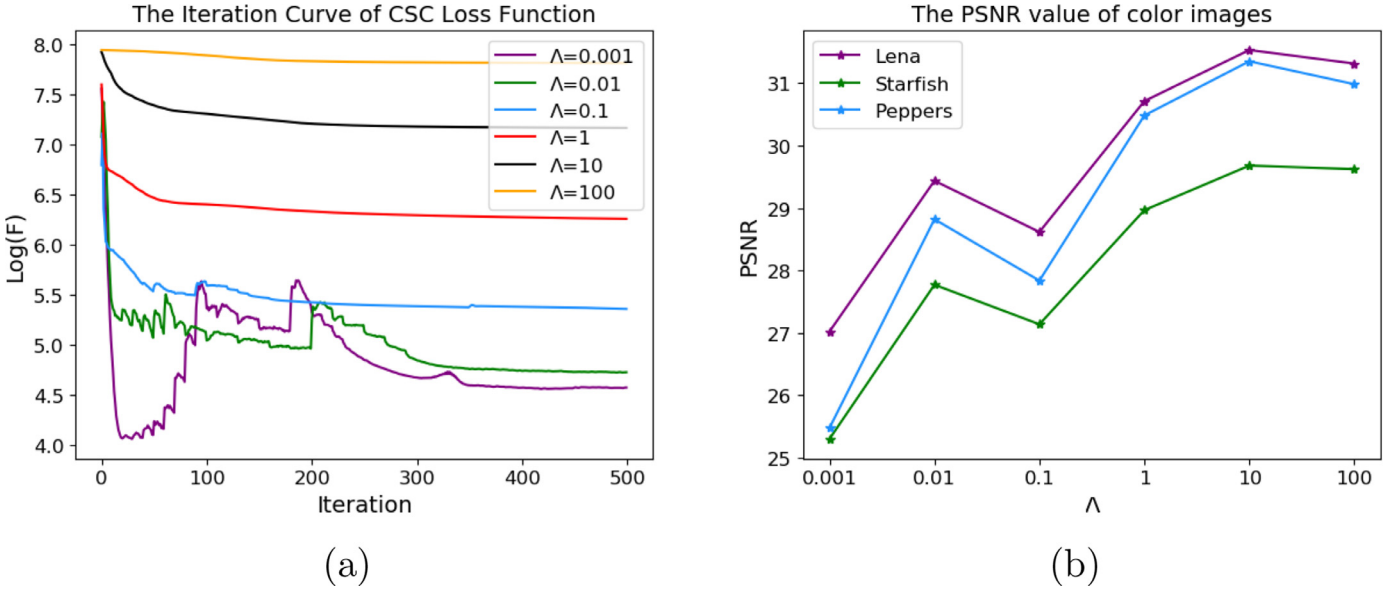


Fig. 5. (a) The loss of CSC on different Λs. (b) The PSNR values of color images recovered by LRTC-CSC-II on different Λs of dictionary for the sampling rate 30%.

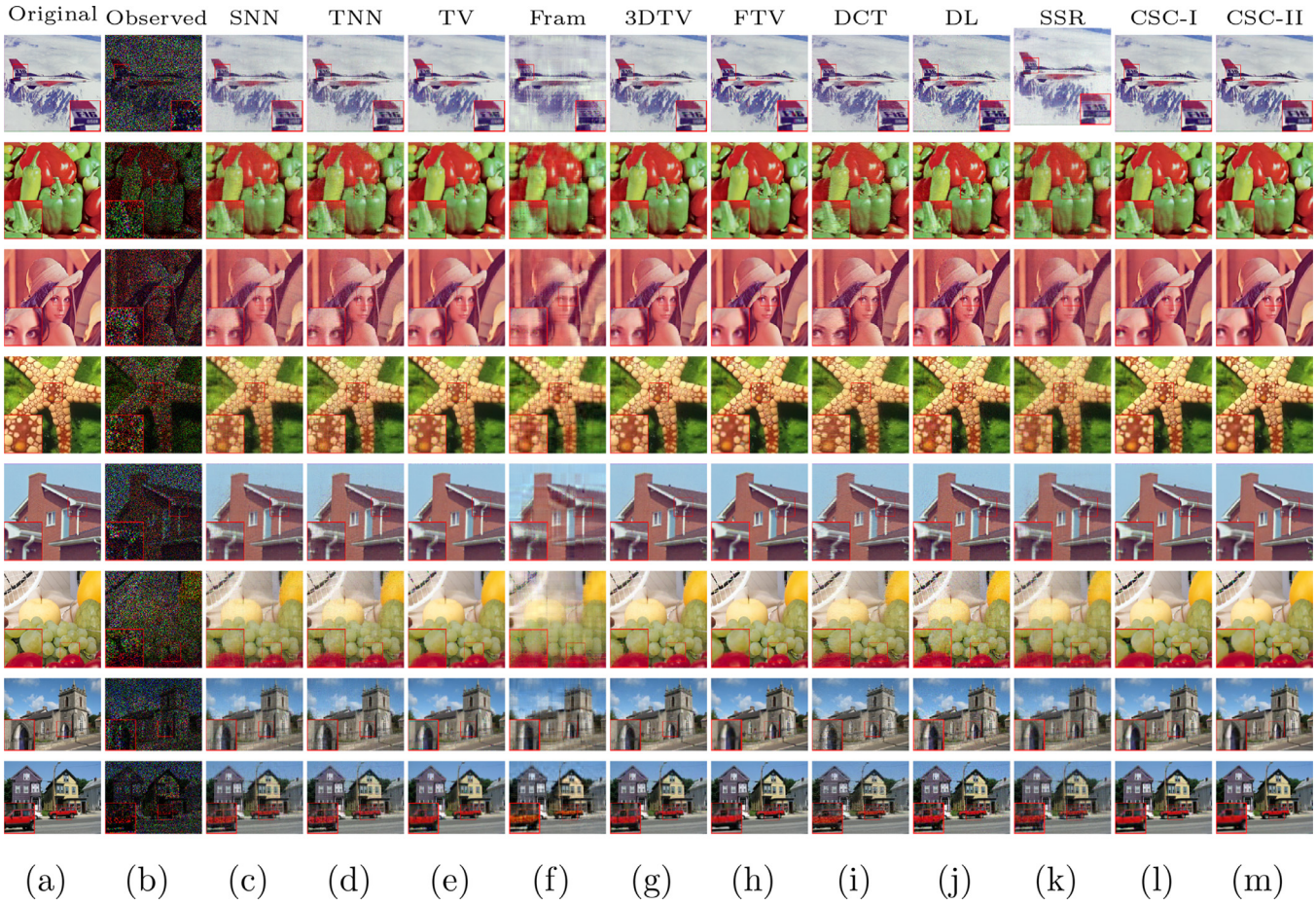


Fig. 6. The recovered color images by different algorithms for the sampling rate 30%, respectively. From (a) to (m): Original image, Observed image, recovered images by SNN, TNN, TV, Framelet(Fram), 3DTV, Framelet-TV(FTV), DCT, WTNNDL(DL), LRSSRTC(SSR), CSC-I and CSC-II, respectively.

Table 1
PSNR results of recovered color images by different algorithms. The **best** and second best values are highlighted in bold and underline, respectively. (MR is short for Missing Ratio).

Image	MR	PSNR										
		SNN	TNN	TV	Fram	3DTV	F-TV	DCT	DL	SSR	CSC-I	CSC-II
Peppers	70%	24.03	23.54	26.87	22.66	27.45	30.43	25.45	25.81	22.36	30.84	**31.72**
256 × 256 × 3	80%	20.99	20.50	24.52	21.63	24.82	28.02	22.74	20.99	20.20	28.20	**29.33**
	90%	16.92	16.75	20.24	18.52	20.61	23.96	19.23	21.81	16.50	23.52	**24.08**
Starfish	70%	23.68	24.41	26.18	22.36	27.27	27.88	26.33	27.25	22.90	28.50	**29.78**
256 × 256 × 3	80%	20.90	21.17	23.55	21.41	24.47	25.18	23.29	24.83	20.81	25.86	**26.75**
	90%	17.29	17.54	20.19	17.90	20.71	21.15	19.68	21.62	17.28	22.20	**22.63**
Lena	70%	25.59	25.86	27.69	22.75	28.72	29.33	27.01	25.95	24.67	30.73	**32.03**
256 × 256 × 3	80%	22.82	23.03	25.84	21.87	26.26	26.99	24.02	24.41	22.92	28.13	**29.00**
	90%	19.26	19.47	22.29	20.59	22.80	23.94	20.52	22.17	19.73	24.58	**25.34**
Fruits	70%	24.08	24.30	26.81	21.15	27.21	28.52	25.58	25.66	23.39	29.44	**30.36**
256 × 256 × 3	80%	21.65	21.79	24.89	20.41	25.07	26.43	23.19	23.96	21.47	26.70	**27.59**
	90%	18.22	18.40	21.69	19.18	21.69	23.54	19.88	21.32	18.12	23.95	**24.26**
House	70%	27.30	27.83	28.45	23.09	30.62	31.82	28.98	26.53	24.86	32.26	**33.00**
256 × 256 × 3	80%	24.22	24.57	26.53	21.81	27.76	29.07	26.31	25.47	23.38	29.81	**30.72**
	90%	20.60	20.62	22.50	20.53	23.78	25.29	22.36	23.12	20.18	25.62	**26.49**
Airplane	70%	25.21	26.30	26.55	21.03	27.68	26.63	26.53	23.17	22.48	28.42	**30.05**
256 × 256 × 3	80%	22.70	24.07	24.13	20.05	25.27	24.42	24.10	21.78	20.89	26.12	**27.37**
	90%	19.54	20.92	20.96	18.94	22.02	21.84	21.07	19.65	18.57	22.98	**23.92**
Church	70%	24.70	25.45	25.83	22.92	27.04	25.68	28.32	26.59	23.84	27.46	**28.68**
256 × 192 × 3	80%	22.30	22.84	23.90	21.90	24.81	23.30	26.02	24.79	22.04	25.54	**26.49**
	90%	18.79	19.33	20.82	20.35	21.69	22.55	20.19	22.04	18.88	22.41	**23.21**
Cars	70%	22.76	23.66	24.18	22.60	25.30	27.18	23.98	25.64	21.94	26.31	**27.41**
256 × 192 × 3	80%	20.15	20.86	21.98	21.23	22.78	24.62	21.63	23.48	19.93	24.07	**24.80**
	90%	16.84	17.60	18.34	16.24	19.52	21.02	18.53	20.41	16.63	20.76	**21.26**

Table 2
SSIM results of recovered color images by different algorithms. The **best** and second best values are highlighted in bold and underline, respectively. (MR is short for Missing Ratio).

Image	MR	SSIM										
		SNN	TNN	TV	Fram	3DTV	F-TV	DCT	DL	SSR	CSC-I	CSC-II
Peppers	70%	0.9339	0.9233	0.9703	0.9147	0.9687	0.9759	0.9496	0.9713	0.9271	0.9855	**0.9865**
256 × 256 × 3	80%	0.8827	0.8580	0.9501	0.8963	0.9450	0.9746	0.9132	0.9553	0.8802	0.9734	**0.9751**
	90%	0.7588	0.7099	0.8906	0.8168	0.8713	0.9362	0.8348	0.9127	0.7539	0.9307	**0.9406**
Starfish	70%	0.9074	0.8986	0.9589	0.8873	0.9574	0.9696	0.9527	0.9618	0.9172	0.9705	**0.9768**
256 × 256 × 3	80%	0.8477	0.8255	0.9272	0.8605	0.9250	0.9504	0.9091	0.9317	0.8721	0.9512	**0.9587**
	90%	0.7320	0.6801	0.8801	0.7323	0.8482	0.9007	0.8177	0.8718	0.7595	0.9013	**0.9074**
Lena	70%	0.9477	0.9460	0.9668	0.9156	0.9722	0.9778	0.9610	0.9652	0.9515	0.9814	**0.9853**
256 × 256 × 3	80%	0.9128	0.9041	0.9511	0.8992	0.9536	0.9642	0.9296	0.9474	0.9237	0.9687	**0.9728**
	90%	0.8450	0.8128	0.9102	0.8684	0.9104	0.9376	0.8677	0.9065	0.8645	0.9398	**0.9436**
Fruits	70%	0.8747	0.8643	0.9475	0.8328	0.9374	0.9619	0.9162	0.9451	0.8959	0.9645	**0.9729**
256 × 256 × 3	80%	0.8174	0.7962	0.9213	0.8053	0.9055	0.9409	0.8657	0.9111	0.8428	0.9433	**0.9495**
	90%	0.7164	0.6575	0.8607	0.7499	0.8294	0.8962	0.7744	0.8392	0.7356	0.8920	**0.9014**
House	70%	0.9233	0.9149	0.9562	0.8593	0.9571	0.9713	0.9510	0.9472	0.9275	0.9717	**0.9745**
256 × 256 × 3	80%	0.8610	0.8406	0.9317	0.8242	0.9244	0.9534	0.9165	0.9262	0.8808	0.9550	**0.9601**
	90%	0.7276	0.6642	0.8611	0.7658	0.8326	0.9009	0.8152	0.8723	0.7489	0.9062	**0.9156**
Airplane	70%	0.6588	0.8859	0.9132	0.6761	0.8104	0.9078	0.8919	0.8632	0.7664	0.9115	**0.9458**
256 × 256 × 3	80%	0.5407	0.8129	0.8580	0.5971	0.7131	0.8613	0.8178	0.7923	0.6754	0.8630	**0.9103**
	90%	0.3668	0.6604	0.7458	0.5090	0.5192	0.7610	0.6685	0.6259	0.5034	0.7678	**0.8184**
Church	70%	0.8327	0.8485	0.8798	0.7922	0.8933	0.9242	0.8822	0.9182	0.8606	0.9130	**0.9313**
256 × 192 × 3	80%	0.7367	0.7497	0.8202	0.7491	0.8271	0.8828	0.8044	0.8734	0.7806	0.8693	**0.8923**
	90%	0.5420	0.5555	0.6959	0.6629	0.6888	0.7835	0.6489	0.7712	0.5956	0.7728	**0.7918**
Cars	70%	0.7817	0.8021	0.8664	0.7889	0.8618	0.9154	0.8723	0.9134	0.8248	0.9058	**0.9218**
256 × 192 × 3	80%	0.6628	0.6780	0.7902	0.7320	0.7717	0.8216	0.7903	0.8627	0.7274	0.8541	**0.8672**
	90%	0.4378	0.4513	0.6106	0.5411	0.5795	0.7354	0.6240	0.7369	0.5163	0.7309	**0.7659**

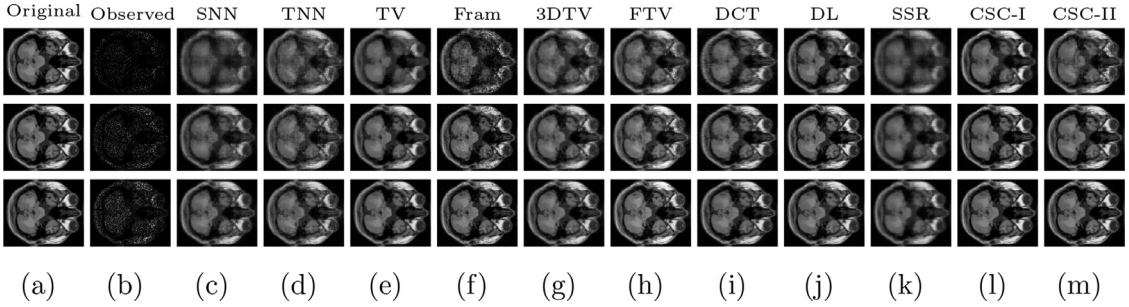


Fig. 7. The 30-th band of recovered MRI data by all algorithms for different sampling rates. From (a) to (m): Original data, Observed data, recovered data by SNN, TNN, TV, Framelet(Fram), 3DTV, Framelet-TV(FTV), DCT, WTNNDL(DL), LRSSRTC(SSR), CSC-I and CSC-II, respectively. From the top down: 10%, 20% and 30% sampling rate.

dictionary used in this part is of size $16 \times 16 \times 30$ for working well in most situations.

Color images own three channels only, while MRI data often include dozen or hundred bands. In this way, the relationship among different bands will be closer, which is very advantageous to the TNN-based method. However, our SNN-based model still shows its superiority after introducing the CSC prior. The same as color images, we randomly pick 10%, 20%, and 30% elements as the observations, and the results are shown in Fig. 7. It's intuitive that LRTC-CSC models achieve the best performance among these models, while LRTC-SNN achieves the worst performance, which verified the effectiveness of CSC regularization. In particular, LRTC-CSC models work well on detailed recovery compared to TNN-3DTV and Framelet-TV. When the missing rate reaches 90%, our model can still hold a relatively clear result.

Fig. 8 further shows the recovery performance in every band. Both the PSNR and SSIM values of each band show the superiority of LRTC-CSC models. 3DTV and DCT methods have close results. Note that TNN-3DTV can achieve a high SSIM value approaching LRTC-CSC. However, there is a gap between them on the edge band of the whole MRI data. The proposed model LRTC-CSC-I achieves a performance promotion of 4 and 0.05 with respect to PSNR and SSIM over the results of TNN-3DTV. Moreover, the performances of LRTC-TNN and LRTC-CSC-II on edge band are similar to TNN-3DTV, which implies the shortcoming of TNN-based methods. In other words, TNN can not handle the edge information due to lacking enough neighbors, which is determined by its predominant characteristic of global information capturing. The performance of TT rank-based method is not satisfactory in MRI data recovering experiments because of imposing KA on the unbalanced tensor. To our surprise, we found that the WTNNDL method performs quite closely to our method LRTC-CSC-I on the MRI dataset, and such results further illustrate the effectiveness of dictionary learning.

4.4. Videos

In this subsection, we benchmarked our model on *Suzie* of size $144 \times 176 \times 150$. The same as MRI experiments, we utilize 30 filters in a dictionary trained by only 10 frames of the video data. And except for the trained ones, another 30 frames of video are used for recovering task. By randomly selecting 10% elements, the

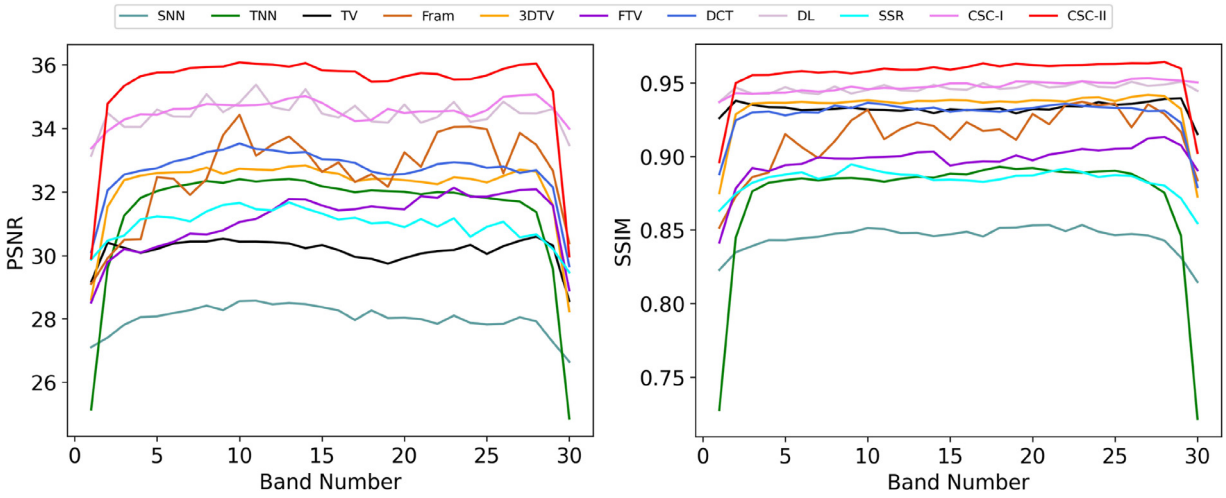


Fig. 8. The PSNR and SSIM values of different bands of MRI data recovered by all algorithms for the sampling rate 30%.

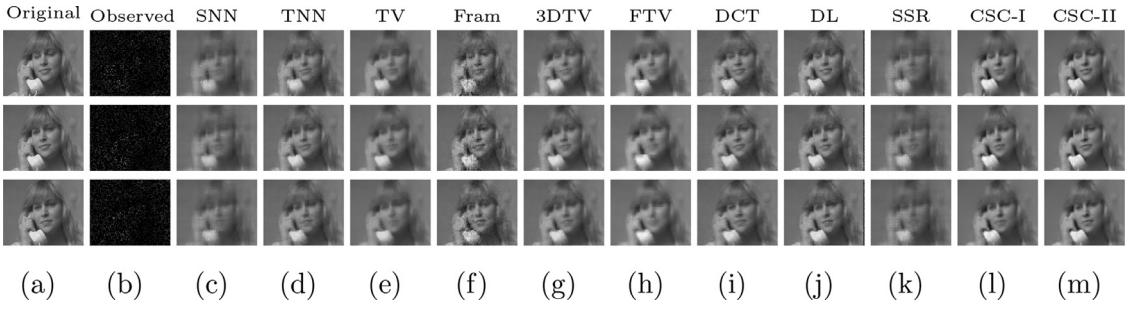


Fig. 9. The 5-th, 15-th and 25-th frame of recovered *suzie* video by all algorithms for 10% sampling rates. From (a) to (m): Original data, Observed data, recovered data by SNN, TNN, TV, Framelet(Fram), 3DTV, Framelet-TV(FTV), DCT, WTNNDL(DL), LRSSRTC(SSR), CSC-I and CSC-II, respectively. From the top down: 5-th, 15-th and 25-th frame.

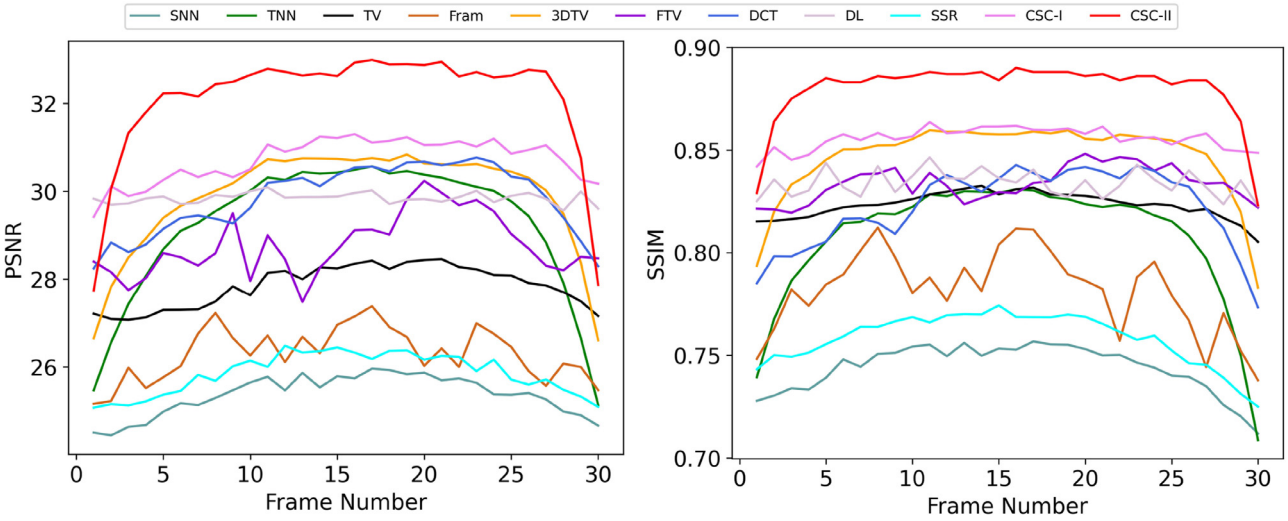


Fig. 10. The PSNR and SSIM values of different frames of video data recovered by all algorithms for the sampling rate 10%.

recovered results are shown in Fig. 9. It is not hard to see that the results recovered by our LRTC-CSC models are the cleanest. Similar to MRI data experiment, TNN-3DTV is close to LRTC-CSC-I but it can not handle the edge frame well, e.g., the 30-th frame in Fig. 10. The performance of the WTNNDL method is degraded compared to the MRI data. It can also be observed that the SNN-based Framelet-TV is inferior to TNN-3DTV, which shows the low-rank approximating ability of TNN.

Fig. 10 shows the recovery results of each frame in detail. Note that LRTC-CSC-II achieves the highest score in both PSNR and SSIM. However, at the beginning or end of the video, only the proposed LRTC-CSC-I is stable while the index curve of the TNN-based methods, including LRTC-CSC-II, drops rapidly, which shows the stability of SNN-based methods on edges, again. Compared with LRTC-SNN and LRTC-TNN, the proposed models have claimed the superiority of CSC regularization.

Table 3
Complexity analysis.

Norm calculation	CSC regularization	
SNN: $\mathcal{O}(\sum_{n=1}^{3}((I_n)^2(\prod_{i\neq n}^{3}I_i)+(\prod_{i\neq n}^{3}I_i)^3))$ TNN: $\mathcal{O}(I_1I_2I_3\log(I_3)+I_{(1)}I_{(2)}^2I_3)$	Fourier transforms $\mathcal{O}(KDlog(D))$	Softthresholding $\mathcal{O}(KD)$

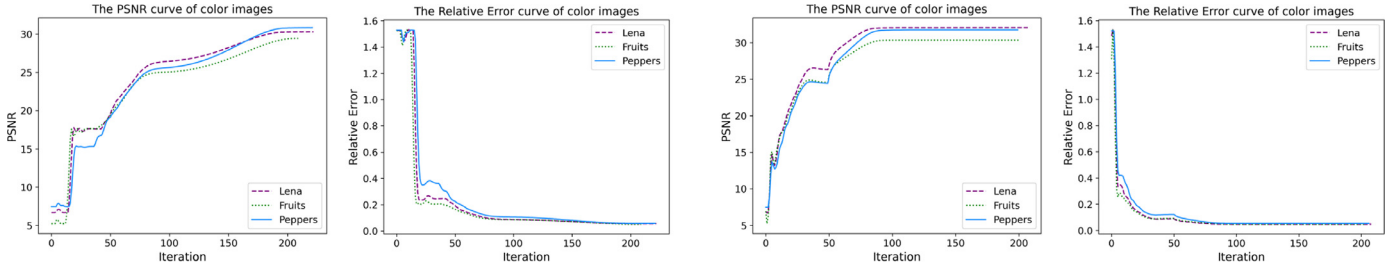


Fig. 11. The convergence behavior with respect to iterations on the color images at $MR = 70\%$. The figures show the convergence of CSC-I (a) and CSC-II (b) algorithms for RE and PSNR, respectively.

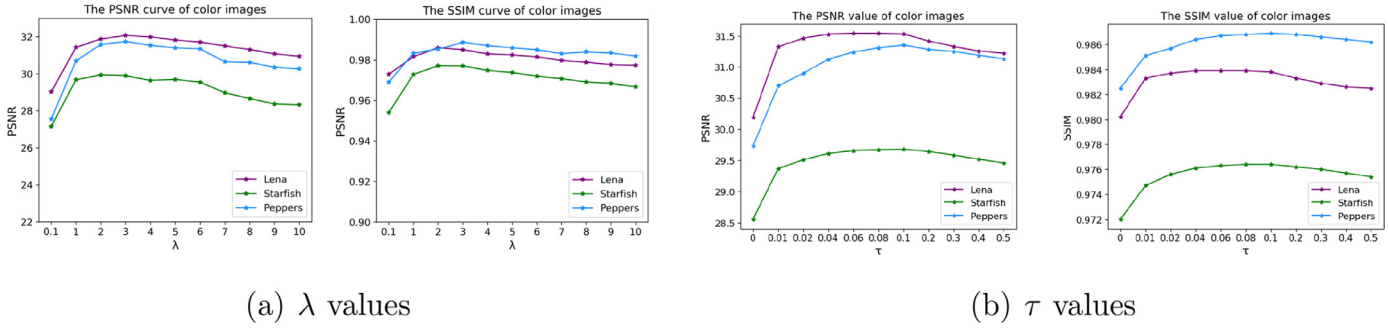


Fig. 12. The PSNR and SSIM values of color images recovered by CSC-II on different λ and τ values for the sampling rate 30%.

4.5. Analysis

4.5.1. Complexity analysis

Here, we briefly analyze the complexity of our methods. in this work the main cost of our approach depends on two parts: training filters and reconstruction. Training filters for CSC using small samples is hardly time consuming. In the reconstruction process, we employ ADMM to solve the optimization problem. It is easy to see that the main cost of each iteration lies in the computation process of the norm and the part of CSC regularization.

For a third-order tensor, $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times I_3}$. We define $I_{(1)} = \max(I_1, I_2)$ and $I_{(2)} = \min(I_1, I_2)$. The calculation complexity of SNN and TNN lies in the Fourier transformation of the matrix and singular value decomposition. Therefore, according to the definition of the two norms, the calculation complexity of SNN is $\mathcal{O}(\sum_{n=1}^{3}((I_n)^2(\prod_{i\neq n}^{3}I_i)+(\prod_{i\neq n}^{3}I_i)^3))$, and the calculation complexity of TNN is $\mathcal{O}(I_1I_2I_3\log(I_3)+I_{(1)}I_{(2)}^2I_3)$. Whereas, the computation complexity of CSC regularization is $\mathcal{O}(KDlog(D)+KD)$, including Fourier transformation and soft threshold calculation, where K is the number of filters and D is the number of samples of a single image. We summarize the above description in Table 3.

Although our method has a computational burden, several techniques including parallel computing and advanced optimization methods can be used to speed up.

4.5.2. Convergency analysis

Recent studies about inexact ADMM show the convergence behavior on different models [34], and we also verify it in our experiments. The numerical experiments have shown the remarkable performance of CSC-I and CSC-II. In order to demonstrate the convergence of algorithms in numerical, Fig. 11 shows the variation curves of the Relative Error (RE) values, i.e., $RE = \frac{\|\mathcal{X}-\mathcal{T}\|_F}{\|\mathcal{T}\|_F}$, and PSNR values of CSC-I and CSC-II algorithms with the number of iterations on three color images of *Lena, Peppers* and *Fruits* at $MR = 70\%$. Our algorithm satisfies the stopping condition when the maximum number of iterations is reached or when RE is less than ϵ ($\epsilon = 1e-5$ in the experiment). Since our method belongs to the non-convex optimization issue, the curve can be observed to become smooth sharply after a short period of fluctuation. Intuitively, the numerical convergence is stable after 100 iterations, with the relative change approaching zero when 200 iterations are reached.

4.6. Parameters analysis

4.6.1. Effect of λ

To investigate the robustness of the regularization coefficients λ, grid search or cross-validation can be considered to search the parameters. We conducted color image experiments for different values of λ, and the results are shown in Fig. 12(a). Initially, the PSNR and SSIM values grow as λ increases, which demonstrates the importance of regularization. As λ continues to rise, the evaluated values decrease only slightly. Therefore, we consider the parameter λ is robust within the common-setting range $\{0.1 \sim 10\}$.

4.6.2. Effect of τ

The extra gradient term in (12) can improve the performance of our models which is suggested in [26], and we verified that in our experiments. To investigate the sensitivity of τ in gradient item, we also perform our model on color images for various τ values. The

results are shown in Fig. 12(b). When $\tau = 0$, our algorithms degenerate to the version without the gradient regularization, and the function of this regularization term is fully reflected in the comparison with the original model. The performance achieves peak when $\tau = 0.1$ in most situations, thus, we set $\tau = 0.1$ in this paper.

5. Conclusion

Inspired by the capacity of CSC on high-frequency component processing, we introduced CSC into the traditional LRTC model. In this way, the details of the underlying tensor would be improved in addition to the low-rank component recovery. Compared to other plug-and-play CNN prior to using thousands of samples, our method can achieve good performance with only small samples. For the proposed models, we obtain effective algorithms based on the inexact ADMM method. And the effectiveness of LRTC-CSC-I and LRTC-CSC-II have been verified in color images, MRI data, and video data recovery experiments. There are still some limitations of our methods. We mainly consider the a priori nature of visual data, thus the models are applicable to visual data and might not be applicable to non-visual data.

In future work, it would be of great interest to leverage the prior information of CSC to improve the performance of other models. For the marginal effects of the TNN-based method, CSC might be a promising method to strengthen the connection between the edge and subject. Meanwhile, exploring a method to automatically find or tune the best parameters, the model will be more applicable and practical. Moreover, we simply display and store the trained dictionaries in random order. Currently, sorting dictionaries before learning sparse representations has also attracted extensive attention. Further research on dictionary sorting is expected to achieve better results.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgment

The research is supported by the National Natural Science Foundation of China (62176269), the Guangdong Basic and Applied Basic Research Foundation (2019A1515011043), and the Innovative Research Foundation of Ship General Performance (25622112).

Appendix A. LRTC-CSC-II

The constraint condition can be rewritten as (6). By introducing auxiliary variables $\mathcal{F} = \mathcal{X}$ and $\mathcal{Z} = \mathcal{X}$, the augmented Lagrangian function of (29) can be formulated as:

$$\begin{aligned}\mathcal{L} = &\sum_{k=1}^{n_3}(\|\hat{F}^{(k)}\|_{*_t} + \frac{\beta_1}{2}\left\|F^{(k)} - X^{(k)}\right\|_F^2 + \left\langle W_1^{(k)}, F^{(k)} - X^{(k)}\right\rangle) \\ &+ \frac{\beta_2}{2}\|\mathcal{Z} - \mathcal{X}\|_F^2 + \langle \mathcal{W}_2, \mathcal{Z} - \mathcal{X}\rangle + P(\mathcal{X}) + \lambda\Phi(\mathcal{Z}).\end{aligned} \tag{A.1}$$

In order to distinguish the k-th frontal slice and the k-th iteration times of a tensor, we use $\mathcal{X}(:,:,k)$ to denote the k-th frontal slice and $\mathcal{X}^k$ for the k-th iteration. We divide the problem (A.1) into three subproblems, i.e. $[\mathcal{F}, \mathcal{Z}]$, $\mathcal{X}$, $\mathcal{W}_1$ and $\mathcal{W}_2$, and solve the problem with the same optimization framework of ADMM.

Algorithm 2 LRTC-CSC-II algorithm.

Input: index set Ω, original data $\mathcal{T}$, filters set $\mathcal{D}$**Parameters**: $\Lambda, \lambda, \rho, \tau, \alpha_k, \omega_1, \omega_2, \beta_1, \beta_2, \epsilon$**Initialization**: $\mathcal{X}_\Omega^0 = \mathcal{T}_\Omega$, $\mathcal{X}_{\bar{\Omega}}^0 = \text{mean}(\mathcal{T})$,
repeat
$\mathcal{F}$**-subproblem**
Update $\mathcal{F}^{i+1}$ by (A.4)
$\mathcal{Z}$**-subproblem**
for $j = 1$ to MaxIter **do**
Update $\mathcal{M}^{j+1}$ by (20)
Update $\mathcal{B}^{j+1}$ by (22)
Update C^{j+1} by (23)
end for
Update $\mathcal{Z}^{i+1}$ by (24)
$\mathcal{X}$**-subproblem**
Update $\mathcal{X}^{i+1}$ by (A.6)
$[\mathcal{W}_1, \mathcal{W}_2]$**-subproblem**
Update $\mathcal{W}_1^{i+1}$ by (A.7)
Update $\mathcal{W}_2^{i+1}$ by (A.8)
until Reach convergence $\|\mathcal{X} - \mathcal{T}\|_F / \|\mathcal{T}\|_F < \epsilon$
Output:The recovered tensor $\mathcal{X}$

Solving $[\mathcal{F}, \mathcal{Z}]$-subproblem

Similarly, we are going to solve the $[\mathcal{F}, \mathcal{Z}]$-subproblem separately, since the variables are decoupled.

1) By fixing $\mathcal{X}$ and $\mathcal{Z}$, rewrite the function of $\mathcal{F}$-subproblem:

$$\begin{aligned}\underset{\mathcal{F}}{\text{argmin}} \quad &\sum_{k=1}^{n_3}\|\hat{\mathcal{F}}(:,:,k)\|_* + \sum_{k=1}^{n_3}\frac{\beta_1}{2}\|\hat{\mathcal{F}}(:,:,k) - \hat{\mathcal{X}}(:,:,k) \\ &+ \frac{\mathcal{W}_1(:,:,k)}{\beta_1}\|_F^2.\end{aligned} \tag{A.2}$$

The iterative closed-form solution of (A.2) is:

$$\begin{aligned}\hat{\mathcal{F}}^{i+1}(:,:,k) &= \mathbf{D}_{\tau_k}\left(\hat{\mathcal{F}}^i(:,:,k) - \frac{\mathcal{W}_1^i(:,:,k)}{\beta_1}\right) \\ &= U\text{diag}(\max(\sigma - \tau_k, 0))V^T,\end{aligned} \tag{A.3}$$

where $\tau_k = \frac{1}{\beta_1}$. After getting $\hat{\mathcal{F}}^{i+1}$, we can get $\mathcal{F}^{i+1}$ by inverse Fourier transform

$$\mathcal{F}^{i+1} = \text{ifft}(\hat{\mathcal{F}}^{i+1}). \tag{A.4}$$

2) By fixing $\mathcal{X}$ and F, the $\mathcal{Z}$-subproblem is:

$$\arg\min_{\mathcal{Z}} \quad \frac{\beta_2}{2}\left\|\mathcal{Z} - (\mathcal{X} - \frac{\mathcal{W}_2}{\beta_2})\right\|_F^2 + \lambda\Phi(\mathcal{Z}),$$

which is the same with Model I. The process of solving $\mathcal{Z}$-subproblem has been introduced in detail in Model I, thus we omit them here for readability.

Solving $\mathcal{X}$-subproblem

By fixing $\mathcal{F}$ and $\mathcal{Z}$, the $\mathcal{X}$-subproblem can be simplified and rewritten as:

$$\begin{aligned}\underset{\mathcal{X}}{\text{argmin}} \quad &\sum_{k=1}^{n_3}\frac{\beta_1}{2}\|\hat{\mathcal{F}}(:,:,k) - \hat{\mathcal{X}}(:,:,k) \\ &+ \frac{\mathcal{W}_1(:,:,k)}{\beta_1}\|_F^2 + \frac{\beta_2}{2}\|\mathcal{Z} - \mathcal{X} + \frac{\mathcal{W}_2}{\beta_2}\|_F^2 + P(\mathcal{X}).\end{aligned} \tag{A.5}$$

The closed-form solution of (A.5) can be obtained by:

$$\mathcal{X}^{i+1} = \left(\frac{\beta_1\mathcal{F}^{i+1} + \beta_2\mathcal{Z}^{i+1} + \mathcal{W}_1^i + \mathcal{W}_2^i}{\beta_1 + \beta_2}\right)_{\bar{\Omega}} + \mathcal{T}, \tag{A.6}$$

where $\bar{\Omega}$ is the complement set of Ω.

Solving $[\mathcal{W}_1, \mathcal{W}_2]$-subproblem

By fixing $\mathcal{X}$ and $\mathcal{F}$, $\mathcal{X}$ and $\mathcal{Z}$, we can update the Lagrangian multiplier ω_1 and ω_2, respectively:

$$\mathcal{W}_1^{i+1} = \mathcal{W}_1^i + \beta_1\left(\mathcal{F}^{i+1} - \mathcal{X}^{i+1}\right), \tag{A.7}$$

and

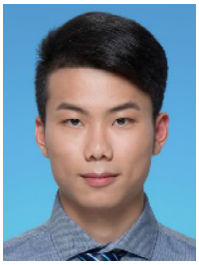
$$\mathcal{W}_2^{i+1} = \mathcal{W}_2^i + \beta_2\left(\mathcal{Z}^{i+1} - \mathcal{X}^{i+1}\right). \tag{A.8}$$

The overall pseudocode of LRTC-CSC-II algorithm is summarized in Algorithm 2. Actually, our model can degenerate to LRTC-TNN model [11] by setting the hyperparameter $\lambda = 0$.

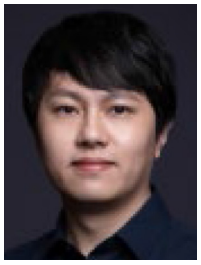
References

[1] H. Lu, K.N. Plataniotis, A.N. Venetsanopoulos, A survey of multilinear subspace learning for tensor data, Pattern Recognit 44 (7) (2011) 1540–1551.
[2] Z. Chen, C. Chen, Z. Zheng, Y. Zhu, Tensor decomposition for multilayer networks clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, 2019, pp. 3371–3378.
[3] M. Yang, Q. Luo, W. Li, M. Xiao, Nonconvex 3d array image data recovery and pattern recognition under tensor framework, Pattern Recognit 122 (2022) 108311.
[4] J. Liu, P. Musialski, P. Wonka, J. Ye, Tensor completion for estimating missing values in visual data, IEEE Trans Pattern Anal Mach Intell 35 (1) (2012) 208–220.
[5] T.G. Kolda, B.W. Bader, Tensor decompositions and applications, SIAM Rev. 51 (3) (2009) 455–500.
[6] J.B. Kruskal, Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics, Linear Algebra Appl 18 (2) (1977) 95–138.
[7] J.A. Bengua, H.N. Phien, H.D. Tuan, M.N. Do, Efficient tensor completion for color image and video recovery: low-rank tensor train, IEEE Trans. Image Process. 26 (5) (2017) 2466–2479.
[8] C. Chen, Z.-B. Wu, Z.-T. Chen, Z.-B. Zheng, X.-J. Zhang, Auto-weighted robust low-rank tensor completion via tensor-train, Inf Sci (Ny) 567 (2021) 100–115.
[9] M.E. Kilmer, K. Braman, N. Hao, R.C. Hoover, Third-order tensors as operators on matrices: atheoretical and computational framework with applications in imaging, SIAM J. Matrix Anal. Appl. 34 (1) (2013) 148–172.
[10] B. Recht, M. Fazel, P.A. Parrilo, Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization, SIAM Rev. 52 (3) (2010) 471–501.
[11] Z. Zhang, G. Ely, S. Aeron, N. Hao, M. Kilmer, Novel methods for multilinear data completion and de-noising based on tensor-SVD, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 3842–3849.
[12] X.L. Zhao, J.H. Yang, T.H. Ma, T.X. Jiang, M.K. Ng, Tensor completion via complementary global, local, and nonlocal priors, IEEE Trans. Image Process. 31 (2022) 984–999.
[13] X. Li, Y. Ye, X. Xu, Low-rank tensor completion with total variation for visual data inpainting, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.
[14] T.-X. Jiang, T.-Z. Huang, X.-L. Zhao, T.-Y. Ji, L.-J. Deng, Matrix factorization for low-rank tensor completion using framelet prior, Inf Sci (Ny) 436 (2018) 403–417.
[15] J. Chen, H. Lin, Y. Shao, L. Yang, Oblique striping removal in remote sensing imagery based on wavelet transform, Int J Remote Sens 27 (8) (2006) 1717–1723.
[16] F. Jiang, X.-Y. Liu, H. Lu, R. Shen, Anisotropic total variation regularized low-rank tensor completion based on tensor nuclear norm for color image inpainting, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2018, pp. 1363–1367.
[17] J. Liu, T.-Z. Huang, I.W. Selesnick, X.-G. Lv, P.-Y. Chen, Image restoration using total variation with overlapping group sparsity, Inf Sci (Ny) 295 (2015) 232–246.
[18] X.-T. Li, X.-L. Zhao, T.-X. Jiang, Y.-B. Zheng, T.-Y. Ji, T.-Z. Huang, Low-rank tensor completion via combined non-local self-similarity and low-rank regularization, Neurocomputing 367 (2019) 1–12.
[19] K. Zhang, W. Zuo, S. Gu, L. Zhang, Learning deep CNN denoiser prior for image restoration, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3929–3938.
[20] M. Elad, M. Aharon, Image denoising via sparse and redundant representations over learned dictionaries, IEEE Trans. Image Process. 15 (12) (2006) 3736–3745.
[21] J. Yang, J. Wright, T.S. Huang, Y. Ma, Image super-resolution via sparse representation, IEEE Trans. Image Process. 19 (11) (2010) 2861–2873.
[22] S.S. Chen, D.L. Donoho, M.A. Saunders, Atomic decomposition by basis pursuit, SIAM Rev. 43 (1) (2001) 129–159.
[23] V. Papyan, Y. Romano, J. Sulam, M. Elad, Convolutional dictionary learning via local processing, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5296–5304.
[24] M.D. Zeiler, D. Krishnan, G.W. Taylor, R. Fergus, Deconvolutional networks, in: 2010 IEEE Computer Society Conference on computer vision and pattern recognition, IEEE, 2010, pp. 2528–2535.
[25] H. Bristow, A. Eriksson, S. Lucey, Fast convolutional sparse coding, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 391–398.
[26] B. Wohlberg, Convolutional sparse representations with gradient penalties, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2018, pp. 6528–6532.
[27] V. Papyan, Y. Romano, M. Elad, Convolutional neural networks analyzed via convolutional sparse coding, The Journal of Machine Learning Research 18 (1) (2017) 2887–2938.
[28] H. Zhang, V.M. Patel, Convolutional sparse and low-rank coding-based rain streak removal, in: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, 2017, pp. 1259–1267.
[29] P. Bao, W. Xia, K. Yang, W. Chen, M. Chen, Y. Xi, S. Niu, J. Zhou, H. Zhang, H. Sun, et al., Convolutional sparse coding for compressed sensing CT reconstruction, IEEE Trans Med Imaging 38 (11) (2019) 2607–2619.
[30] A. Bibi, B. Ghanem, High order tensor formulation for convolutional sparse coding, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 1772–1780.
[31] R. Xu, Y. Xu, Y. Quan, Factorized tensor dictionary learning for visual tensor data completion, IEEE Trans Multimedia (2020).
[32] X.-L. Zhao, W.-H. Xu, T.-X. Jiang, Y. Wang, M.K. Ng, Deep plug-and-play prior for low-rank tensor completion, Neurocomputing (2020).
[33] K. Zhang, W. Zuo, L. Zhang, FFDNet: toward a fast and flexible solution for CNN-based image denoising, IEEE Trans. Image Process. 27 (9) (2018) 4608–4622.
[34] W.W. Hager, H. Zhang, Inexact alternating direction methods of multipliers for separable convex optimization, Comput Optim Appl 73 (1) (2019) 201–235.
[35] M.E. Kilmer, C.D. Martin, Factorization strategies for third-order tensors, Linear Algebra Appl 435 (3) (2011) 641–658.
[36] C. Chen, M.K. Ng, X.-L. Zhao, Alternating direction method of multipliers for nonlinear image restoration problems, IEEE Trans. Image Process. 24 (1) (2014) 33–43.
[37] G.H. Golub, P.C. Hansen, D.P. O'Leary, Tikhonov regularization and total least squares, SIAM J. Matrix Anal. Appl. 21 (1) (1999) 185–194.
[38] B. Wohlberg, Efficient algorithms for convolutional sparse representations, IEEE Trans. Image Process. 25 (1) (2015) 301–315.
[39] J.-H. Yang, X.-L. Zhao, T.-H. Ma, M. Ding, T.-Z. Huang, Tensor train rank minimization with hybrid smoothness regularization for visual data recovery, Appl Math Model 81 (2020) 711–726.
[40] C. Lu, X. Peng, Y. Wei, Low-rank tensor completion with a new tensor nuclear norm induced by invertible linear transforms, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 5996–6004.
[41] Y. Du, G. Han, Y. Quan, Z. Yu, H.-S. Wong, C.L.P. Chen, J. Zhang, Exploiting global low-rank structure and local sparsity nature for tensor completion, IEEE Trans Cybern 49 (11) (2018) 3898–3910.
[42] X. Chen, S. Liang, Z. Zhang, F. Zhao, A novel spatio-temporal data low-rank imputation approach for traffic sensor network, IEEE Internet Things J. (2022).

Tianchi Liao received the B.S. degree majoring in Information and Computing Science in Sichuan Agricultural University, Sichuan, China, in 2021. And now she is pursuing the Master degree in Sun Yat-sen University. Her research interests include models and algorithms of low-rank modeling for image processing problems.



Zhebin Wu received the B.S. degree majoring in Information and Computing Science in Sun Yat-sen University, Guangzhou, China, in 2020. And now he is pursuing the Master degree in Sun Yat-sen University. His current research interests focus on numerical optimization, numerical linear algebra and machine learning.



Chuan Chen received the B.S. degree from SunYat-sen University, Guangzhou, China, in 2012, andthe Ph.D. degree from Hong Kong Baptist University, Hong Kong, in 2016. He is currently a Research Associate Professor with the School of Data and Computer Science, SunYat-Sen University. His current research interestsinclude machine learning, numerical linear algebra,and numerical optimization.

Zibin Zheng received the Ph.D. degree from the received the Ph.D. degree from the Chinese University of Hong Kong, in 2011. He is currently a Professor at School of Data and Computer Science with Sun Yat-sen University, China. He serves as Chairman of the Software Engineering Department. He published over 120 international journal and conference papers, including 3 ESI highlycited papers. According to Google Scholar, his papers have more than 7000 citations, with an H-index of 42. His research interests include blockchain, services computing, software engineering, and financial big data. He was a recipient of several awards, including the Top 50 Influential Papers in Blockchain of 2018, the ACM SIGSOFT Distinguished Paper Award at ICSE2010, the Best Student Paper Award at ICWS2010. He served as BlockSys'19 and CollaborateCom'16 General Co-Chair, SC2'19, ICIOT'18 and IoV'14 PC CoChair.

Xiongjun Zhang received the Ph.D. degree from the College of Mathematics and Econometrics, Hunan University, Changsha, China, in 2017. From 2015 to 2016, he was an Exchange Ph.D. Student with the Department of Mathematics, Hong Kong Baptist University, Hong Kong. He is currently an Assistant Professor with the School of Mathematics and Statistics, Central China Normal University, Wuhan, China. His current research interests include image processing and tensor optimization.