Tensor Completion Network for Visual Data

Xiang-Yu Wang, Xiao-Peng Li, *Member, IEEE*, Nicholas D. Sidiropoulos, *Fellow, IEEE*, and Hing Cheung So, *Fellow, IEEE*

***Abstract*—Tensor completion aims at filling in the missing elements of an incomplete tensor based on its partial observations, which is a popular approach for image inpainting. Most existing methods for visual data recovery can be categorized into traditional optimization-based and neural network-based methods. The former usually adopt a low-rank assumption to handle this ill-posed problem, enjoying good interpretability and generalization. However, as visual data are only approximately low rank, handcrafted low-rank priors may not capture the complex details properly, limiting the recovery performance. For neural network-based methods, despite their impressive performance in image inpainting, sufficient training data are required for parameter learning, and their generalization ability on the unseen data is a concern. In this paper, combining the advantages of these two distinct approaches, we propose a tensor Completion neural Network (CNet) for visual data completion. The CNet is comprised of two parts, namely, the encoder and decoder. The encoder is designed by exploiting the CANDECOMP/PARAFAC decomposition to produce a low-rank embedding of the target tensor, whose mechanism is interpretable. To compensate the drawback of the low-rank constraint, a decoder consisting of several convolutional layers is introduced to refine the low-rank embedding. The CNet only uses the observations of the incomplete tensor to recover its missing entries and thus is free from large training datasets. Extensive experiments in inpainting color images, grayscale video sequences, hyperspectral images, color video sequences, and light field images are conducted to showcase the superiority of CNet over state-of-the-art methods in terms of restoration performance.**

***Index Terms*—Tensor completion, CP decomposition, unrolling, encoder-decoder neural network, image inpainting, visual data recovery.**

Received 26 October 2023; revised 30 July 2024 and 3 November 2024; accepted 5 December 2024. Date of publication 1 January 2025; date of current version 16 January 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62401373, in part by the Young Innovative Talents Project of Guangdong Provincial Department of Education (Natural Science) under Grant 2023KQNCX063, and in part by the Project of Department of Education of Guangdong Province under Grant 2021KCXTD008. The work of Nicholas D. Sidiropoulos was not supported by any grant or sponsor. The associate editor coordinating the review of this article and approving it for publication was Samet Oymak. *(Corresponding author: Xiao-Peng Li.)*

Xiang-Yu Wang and Hing Cheung So are with the Department of Electrical Engineering, City University of Hong Kong, Hong Kong SAR, China (e-mail: xwang2286-c@my.cityu.edu.hk; hcso@ee.cityu.edu.hk).

Xiao-Peng Li is with the State Key Laboratory of Radio Frequency Heterogeneous Integration, Shenzhen University, Shenzhen 518060, China (e-mail: x.p.li@szu.edu.cn).

Nicholas D. Sidiropoulos is with the Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904 USA (e-mail: nikos@virginia.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TSP.2024.3524568, provided by the authors.

Digital Object Identifier 10.1109/TSP.2024.3524568

I. Introduction

TENSORS, which are multi-way arrays indexed by more than two indices, can be regarded as a generalization of matrices. They depict the multi-dimensional structures naturally and hence have been widely used to model real-world data. For example, color images, comprised of red, green, and blue (RGB) channels, can be represented by third-order tensors. A hyperspectral image (HSI) can also be modeled as a three-way tensor indexed by two spatial dimensions and one spectral band. Similarly, videos can be expressed as multi-dimensional tensors indexed by spatial coordinates, RGB channels, and temporal frames. As a result, tensor analysis has impacted a wide range of research areas and applications, such as data mining [1], [2], pattern recognition [3], [4], and computer vision [5], [6]. In these fields, tensor completion (TC) has emerged as an important and popular topic, serving as the foundation for numerous methods in image and video inpainting [7], [8], [9], [10].

TC aims to restore the missing elements of a tensor using its available elements, which is a highly ill-posed problem without additional constraints, and hence a low-rank assumption is usually adopted. Tensor rank and multilinear (mode) ranks have been proposed for tensor analysis. Under the corresponding low-rank constraints, various optimization strategies have been employed to handle TC, and traditional optimization-based TC approaches can be classified as rank-minimization-based and factorization-based methods. The CANDECOMP/PARAFAC (CP) rank of a tensor $\mathcal{X}$ is defined as the smallest number of rank-1 tensors whose sum produces $\mathcal{X}$ [11], [12], [13]. This definition generalizes the notion of matrix rank in view of the outer product representation. However, determining the CP rank of a given tensor has been proven to be NP-hard [14], and minimizing the CP rank is almost impossible. Hence, few rank-minimization-based TC algorithms adopt the CP rank. Instead, there are TC methods based on CP decomposition under the assumption that (an upper bound to) the CP rank is known [15], [16]. Moreover, Zhao et al. [17] propose a fully Bayesian CP method utilizing prior information for performance improvement. Another idea using prior information is smooth PARAFAC TC [18].

A direct way to handle TC is to convert a tensor to multiple matrices with the same number of elements, which is called matricization. The n-Rank [13], also known as the multilinear rank, is a vector whose coefficients are the ranks of the unfolded tensors along each mode. For an Nth-order tensor $\mathcal{X}$, its n-Rank is defined as

$$\text{rank}_{\text{Tucker}}(\mathcal{X}) = [\text{rank}(\mathbf{X}_{(1)}), \text{rank}(\mathbf{X}_{(2)}), \cdots, \text{rank}(\mathbf{X}_{(N)})], \tag{1}$$

where $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times \prod_{k=1,k\neq n}^{N} I_k}$ is the mode-n unfolding of $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}$ [13]. The n-Rank is related to Tucker decomposition and widely used to tackle the Tucker rank involved problems. TC based on the Tucker rank minimization is formulated as:

$$\min_{\mathcal{X}} \sum_{n=1}^{N} \alpha_n \text{rank}(\mathbf{X}_{(n)}) \quad \text{s.t. } \mathfrak{P}_\Omega(\mathcal{X}) = \mathfrak{P}_\Omega(\mathcal{Y}), \tag{2}$$

where $\mathcal{Y}$ represents the given tensor with partially observed elements, $\mathcal{X}$ is the target tensor, and $\mathfrak{P}_\Omega(\cdot)$ is an operator extracting elements whose indices belong to the observation set Ω. It is required that $\sum_{n=1}^{N} \alpha_n = 1$. To solve (2), Liu et al. introduce tensor nuclear norm by generalizing the matrix trace norm and propose SiLRTC, HaLRTC, and FaLRTC [7]. Besides, TMac [19] employs low-rank matrix factorization to all the unfolded matrices and utilizes matrix completion to perform TC. However, under unbalanced unfolding scenarios, the unfolded matrices might be too "fat", that is, the number of columns is much greater than the number of rows, especially for high-order tensors. Therefore, rank of the unfolded matrix is limited by the row and column numbers, and it may not fully capture the global information contained within the tensor [8].

To address this issue, a more even unfolding, viz. n-unfolding, has been proposed [20], which converts $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}$ into $\mathbf{X}_{[n]} \in \mathbb{R}^{\prod_{k=1}^{n} I_k \times \prod_{k=n+1}^{N} I_k}$. The ranks of the n-unfolding matrices align with the tensor train (TT) rank [8], [21]. Based on TT rank, SiLRTC-TT and TMac-TT [8] have been developed for image inpainting. Nonetheless, TT decomposition is sensitive to the rearrangement of tensor dimensions. To mitigate this disadvantage, tensor ring (TR) decomposition is proposed in [22], which represents a high-dimensional tensor as a circular product of a sequence of third-order tensors, indicating that all modes are treated equivalently. TC methods based on TR decomposition can be found in [23], [24], [25].

Nevertheless, direct matricization reorganizes tensors and may distort the original data structure [26], [27], which can introduce artifacts into the results. Unlike multilinear ranks [28], tensor tubal rank is defined as the number of nonzero singular tubes of $\mathcal{S}$, which comes from the t-SVD of $\mathcal{X}$, namely, $\mathcal{X} = \mathcal{U} \star \mathcal{S} \star \mathcal{V}^T$ [29]. Here, $\star$ denotes the t-product [30]. Algorithms based on tensor tubal rank minimization for TC are presented in [28], [31]. According to [28], minimizing tensor tubal rank of $\mathcal{X}$ is equal to minimizing the nuclear norm of the block diagonal matrix whose diagonal elements are the frontal slices of the Fourier transform of $\mathcal{X}$. Other invertible linear transforms are also adopted to calculate the tubal rank in [32], [33]. While satisfactory results can be achieved with an appropriately chosen transform, there are no guidelines for selecting the optimal linear transform. Employing the t-product, Zhou et al. [26] propose a tensor-factorization-based TC algorithm and provide a scheme to automatically determine the tensor tubal rank.

All the aforementioned algorithms exploit traditional optimization theories to deal with the low-rank TC problem, which enjoys good interpretability and generalization. However, the handcrafted low-rank prior cannot capture the complex detail information well [34], resulting in degraded restoration. To improve the performance, there are algorithms [35], [36] combining the plug-and-play denoiser [37] with the low-rankness, where the optimization problem is iteratively solved.

In the last few decades, neural network-based inpainting algorithms [38] have exhibited impressive results for TC. However, most prevailing methods require large datasets for training. A network trained using specific datasets may not perform well on the unseen datasets. Thus, the generalization ability of neural network-based methods is a major concern. There are also unsupervised algorithms, like deep image prior (DIP) [39]. They are designed to handle single image inpainting and do not require extra training data, achieving both satisfying recovery performance and good generalization. Notably, one prominent drawback for DIP is overfitting, and the number of iterations should be carefully chosen. Besides, for most learning-based methods, the "black box" mechanism hinders model interpretability. That is, the desired parameters are acquired by data-driven training, and correct recovery is also not theoretically guaranteed.

Recently, unrolling[1] has attracted considerable attention in the neural network design [40], which maps an iterative algorithm to a deep neural network. In this type of neural networks, the meaning of the data flow is clear, allowing both interpretability and effectiveness in the optimization and learning aspects, respectively. In this work, inspired by unrolling concept, a tensor **C**ompletion neural **Net**work, named CNet, is devised for visual data recovery. We build CNet as an encoder-decoder architecture. Given a tensor, the encoder is designed by exploiting CP decomposition to construct its low-rank approximation [11]. Low-rank TC methods have been proven to be able to recover an incomplete tensor [41], [42], [43]. Therefore, there are theoretical foundations for CNet to restore the missing elements. On the other hand, visual data correspond to approximately low rank, indicating that the low-rank constraint may result in degraded or even heavily distorted detail information. To compensate this drawback, a decoder composed of convolutional neural networks (CNN) is utilized to restore the complex details. Given an incomplete image or video, the CNet only uses its observations to recover the missing entries and thus is unsupervised, where both the encoder and decoder are **not pretrained**. As shown in Fig. 1, the proposed CNet takes advantage of both optimization-based and learning-based

[1]"Unrolling" is also known as "unfolding" in designing neural networks. In fact, these two words are also used in the literature involving tensor matricization, e.g., "unfolding/unrolling a tensor". To avoid confusion, in this paper, "unfolding" refers to the matricization of a tensor, and "unrolling" means the technique of neural network design.

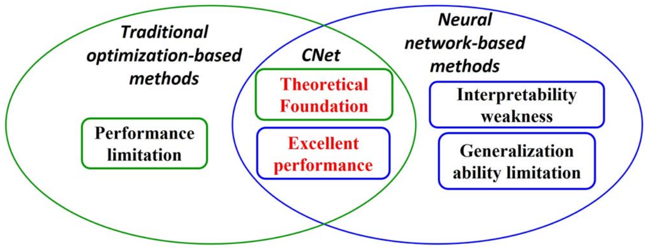
Traditional optimization-based methods
Performance limitation
CNet
Theoretical Foundation
Excellent performance
Neural network-based methods
Interpretability weakness
Generalization ability limitation

Fig. 1. Advantages of CNet compared over traditional optimization-based TC methods and neural network-based TC methods for visual data.

methods. The main contributions of this paper are summarized as follows:

(1) To the best of our knowledge, CNet is the first approach to handle the approximate low-rank TC problem via recovery of the low-rank latent and subsequent enhancement of the fine details using neural networks. Specifically, the incomplete tensor is first restored as a low-rank approximation and further refined to improve the recovery performance, where both steps employ neural networks.

(2) We exploit the low-rank CP decomposition to design our encoder, such that the encoder mechanism is interpretable. Based on CP decomposition, the encoder constructs a low-rank approximation, which is a low-dimensional image embedding and acts as a coarse anchor of the final output.

(3) We introduce a decoder for low-rank TC, which compensates the drawback of applying a low-rank model to images or videos. Since the visual data are only approximately low rank, our recovery performance is superior to those of approaches based on the low-rank assumption.

(4) Extensive experimental results show that the CNet outperforms the state-of-the-art TC algorithms in color image, grayscale video, HSI, color video, and light field image inpainting. A preview of recovery result of light field image *Lego Truck* when only 5% elements are observed is shown in Fig. 2.

The rest of this paper is organized as follows. In Section II, we introduce notation and the CP decomposition theory. The TC problem is then formulated. The design and structure of CNet are presented in Section III. Further investigation and discussion on CNet are given in Section IV. Evaluation of the CNet using visual data is provided in Section V to demonstrate its superior performance over existing methods. Section VI draws our conclusions.

II. Notation and Problem Formulation

A. Notation

The order of a tensor refers to the number of its indices, dimensions, or modes. In this paper, scalars, vectors, matrices, and tensors (number of modes $n \geq 3$) are denoted by lowercase or capital letters, boldface lowercase letters, boldface capital letters, and calligraphic capital letters, respectively. The ith entry of a vector $\mathbf{x}$ is denoted by x_i. The (i_1, i_2) element of a matrix $\mathbf{X}$ is x_{i_1,i_2}. The ith column vector of $\mathbf{X}$ is denoted

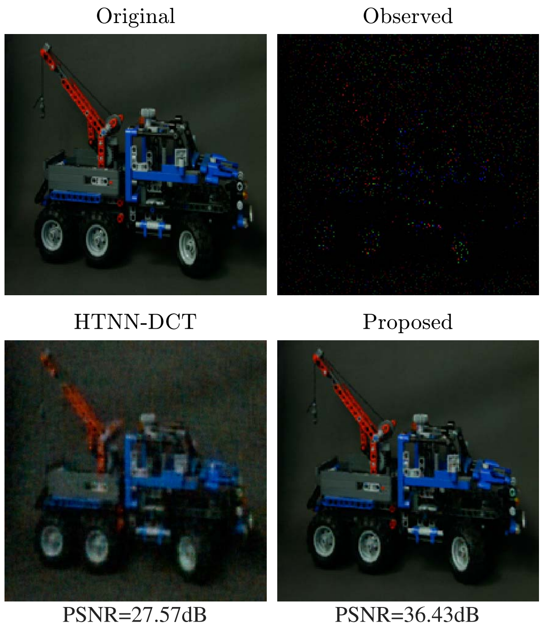

Fig. 2. Sneak preview of recovered light field image *Lego Truck* when only 5% elements are observed.

by $\mathbf{x}_i$. We use $\mathbf{A}^{(n)}$ to represent the nth matrix in a matrix sequence, and $\mathbf{a}^{(n)}$ is defined similarly. For an Nth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, its $(i_1, i_2, \cdots, i_N)$ element is denoted by $x_{i_1,i_2,\cdots,i_N}$. The inner product of two tensors with the same dimensions and the Frobenius norm are, respectively, defined as:

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1,i_2,\cdots,i_N} a_{i_1,i_2,\cdots,i_N} b_{i_1,i_2,\cdots,i_N}, \tag{3a}$$

$$\|\mathcal{A}\|_F = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}. \tag{3b}$$

The N-fold contraction of a set of vectors, matrices or tensors is equal to the sum of their element-wise products. For instance, given a set of matrices $\{\mathbf{A}^{(n)} | n = 1, \cdots, N\}$, N-fold contraction is:

$$\left\langle \mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \cdots, \mathbf{A}^{(N)} \right\rangle = \sum_{i_1,i_2} a^{(1)}_{i_1,i_2} a^{(2)}_{i_1,i_2} \cdots a^{(N)}_{i_1,i_2}. \tag{4}$$

B. CP Decomposition

An Nth-order tensor is said to be a rank-one tensor if it can be represented as the outer product of N vectors [13]:

$$\mathcal{X} = \mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(N)}. \tag{5}$$

The CP decomposition factorizes a tensor as the sum of rank-one tensors, and the minimum number of rank-one tensors needed to produce a tensor is defined as its (CP) rank [13]. Given a third-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ of rank at most F, it can be decomposed into:

$$\mathcal{X} = \sum_{r=1}^{F} \mathbf{a}_r^{(1)} \circ \mathbf{a}_r^{(2)} \circ \mathbf{a}_r^{(3)} = [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}]\!], \tag{6}$$

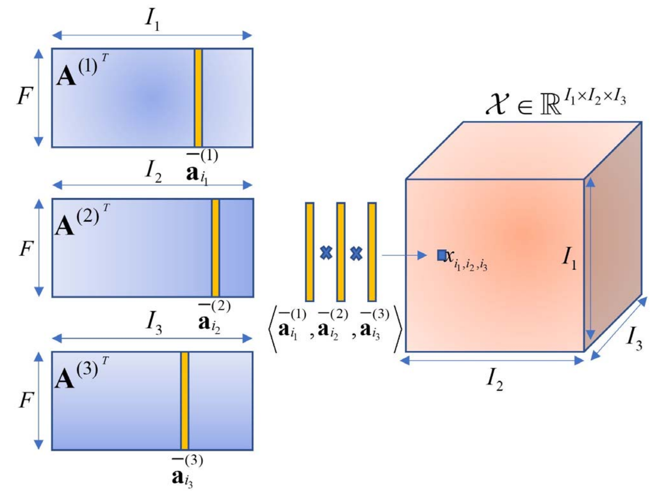
I_1
F
$\mathbf{A}^{(1)^T}$
$\bar{\mathbf{a}}_{i_1}^{(1)}$
I_2
$\mathbf{A}^{(2)^T}$
$\bar{\mathbf{a}}_{i_2}^{(2)}$
I_3
$\mathbf{A}^{(3)^T}$
$\bar{\mathbf{a}}_{i_3}^{(3)}$
$\langle \bar{\mathbf{a}}_{i_1}^{(1)}, \bar{\mathbf{a}}_{i_2}^{(2)}, \bar{\mathbf{a}}_{i_3}^{(3)} \rangle$
$\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$
x_{i_1,i_2,i_3}
I_1
I_2
I_3

Fig. 3. Illustration of CP decomposition for a third-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ with rank F. Here, we use the N-fold contraction among the column vectors of $\mathbf{A}^{(1)^T}$, $\mathbf{A}^{(2)^T}$, and $\mathbf{A}^{(3)^T}$ to calculate elements of $\mathcal{X}$.

where $[\![\cdot]\!]$ is the Kruskal operator and $\mathbf{a}_r^{(n)} \in \mathbb{R}^{I_n}$ for $n = 1, 2, 3$. The factor matrix $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times F}$ represents the concatenation of column vectors $\mathbf{a}_r^{(n)}$, i.e.,

$$\begin{aligned}\mathbf{A}^{(n)} &= \left[\mathbf{a}_1^{(n)}, \cdots, \mathbf{a}_r^{(n)}, \cdots, \mathbf{a}_F^{(n)}\right] \\ &= \left[\bar{\mathbf{a}}_1^{(n)}, \cdots, \bar{\mathbf{a}}_{i_n}^{(n)}, \cdots, \bar{\mathbf{a}}_{I_n}^{(n)}\right]^T. \end{aligned} \tag{7}$$

Here, $\bar{\mathbf{a}}_{i_n}^{(n)} \in \mathbb{R}^F$ is the ith column vector of $\mathbf{A}^{(n)^T}$.

We illustrate the CP decomposition for a third-order tensor in Fig. 3. There are several methods available for computing the CP decomposition with a specified rank, and one widely used algorithm is alternating least squares (ALS) [12].

C. Tensor Completion Formulation

Let $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ be a tensor with partially observed entries $\{y_{i_1,i_2,i_3} | (i_1, i_2, i_3) \in \Omega\}$ under a given index set Ω. TC aims at estimating the unobserved elements in $\mathcal{Y}$. Using the low-rank assumption, rank-minimization-based TC is formulated as [7]:

$$\mathcal{X}^* = \arg\min_{\mathcal{X}} \ \text{rank}(\mathcal{X}), \quad \text{s.t. } \mathfrak{P}_\Omega(\mathcal{X}) = \mathfrak{P}_\Omega(\mathcal{Y}). \tag{8}$$

Note that here rank$(\cdot)$ can be CP rank or multilinear ranks.

Based on factorization, low-rank TC can be posed as [7]:

$$\begin{aligned}\mathcal{X}^* &= \arg\min_{\mathcal{X}} \ \|\mathfrak{P}_\Omega(\mathcal{X}) - \mathfrak{P}_\Omega(\mathcal{Y})\|_F^2, \\ &\text{s.t. } \mathcal{X} = \text{RC}(\mathcal{F}_1, \mathcal{F}_2, \cdots, \mathcal{F}_{\text{sup}}),\end{aligned} \tag{9}$$

where $\mathcal{F}_1, \mathcal{F}_2, \cdots, \mathcal{F}_{\text{sup}}$[2] are the factors of $\mathcal{X}$ by a certain tensor decomposition, and RC$(\cdot)$ represents the corresponding tensor reconstruction operator.

[2]These factors can be vectors, matrices or tensors. Here, we just use tensor notions for ease of representation.

III. Proposed Neural Network

In this section, for visual data completion, we develop an effective encoder-decoder network based on CP decomposition. The encoder constructs a tensor in a low-rank domain by Kruskal operation. The decoder further processes the encoder output and refines its details to improve recovery performance.

A. Algorithm Development

Utilizing the conventional optimization strategy, TC based on low-rank CP decomposition with specified CP rank R is formulated as:

$$\begin{aligned}\min_{\mathbf{A}^{(1)},\mathbf{A}^{(2)},\mathbf{A}^{(3)}} \ &\|\mathfrak{P}_\Omega(\mathcal{X}) - \mathfrak{P}_\Omega(\mathcal{Y})\|_F^2, \\ &\text{s.t. } \mathcal{X} = [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}]\!],\end{aligned} \tag{10}$$

where $\mathcal{X}$, $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ are third-order tensors, and $\mathbf{A}^{(1)} \in \mathbb{R}^{I_1 \times R}$, $\mathbf{A}^{(2)} \in \mathbb{R}^{I_2 \times R}$, and $\mathbf{A}^{(3)} \in \mathbb{R}^{I_3 \times R}$ are the factor matrices.

We rewrite (10) in the element-wise form:

$$\begin{aligned}\min_{\mathbf{A}^{(1)},\mathbf{A}^{(2)},\mathbf{A}^{(3)}} \sum_{(i_1,i_2,i_3)\in\Omega} &\left(\mathfrak{P}_{\{(i_1,i_2,i_3)\}}(\mathcal{X}) - \mathfrak{P}_{\{(i_1,i_2,i_3)\}}(\mathcal{Y})\right)^2, \\ &\text{s.t. } \mathcal{X} = [\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}]\!].\end{aligned} \tag{11}$$

According to (6), it follows that

$$\mathfrak{P}_{\{(i_1,i_2,i_3)\}}(\mathcal{X}) = x_{i_1,i_2,i_3} = \left\langle \bar{\mathbf{a}}_{i_1}^{(1)}, \bar{\mathbf{a}}_{i_2}^{(2)}, \bar{\mathbf{a}}_{i_3}^{(3)} \right\rangle. \tag{12}$$

That is, the (i_1, i_2, i_3) entry is calculated as the N-fold contraction of the i_1th, i_2th, and i_3th column vectors of $\mathbf{A}^{(1)^T}$, $\mathbf{A}^{(2)^T}$, and $\mathbf{A}^{(3)^T}$, respectively, given by

$$x_{i_1,i_2,i_3} = \left\langle \mathbf{A}^{(1)^T} \mathbf{e}_{i_1}, \mathbf{A}^{(2)^T} \mathbf{e}_{i_2}, \mathbf{A}^{(3)^T} \mathbf{e}_{i_3} \right\rangle, \tag{13}$$

where $\mathbf{e}_{i_1} \in \mathbb{R}^{I_1}$ represents a column vector whose entries are all zeros except that the i_1th entry is one, while $\mathbf{e}_{i_2}$ and $\mathbf{e}_{i_3}$ are defined similarly. That is to say, $\mathbf{e}_{i_1}$, $\mathbf{e}_{i_2}$, and $\mathbf{e}_{i_3}$ indicate the locations of x_{i_1,i_2,i_3} in $\mathcal{X}$ and the corresponding column vectors in $\mathbf{A}^{(1)^T}$, $\mathbf{A}^{(2)^T}$, and $\mathbf{A}^{(3)^T}$. Therefore, they are referred to as index vectors. Then, we further express (11) as:

$$\begin{aligned}&\min_{\mathbf{A}^{(1)},\mathbf{A}^{(2)},\mathbf{A}^{(3)}} \sum_{(i_1,i_2,i_3)\in\Omega} \\ &\times \left(\left\langle \mathbf{A}^{(1)^T} \mathbf{e}_{i_1}, \mathbf{A}^{(2)^T} \mathbf{e}_{i_2}, \mathbf{A}^{(3)^T} \mathbf{e}_{i_3} \right\rangle - y_{i_1,i_2,i_3}\right)^2.\end{aligned} \tag{14}$$

Next, we show how to design the encoder by mapping (14) into a neural network. Specifically, $\mathbf{A}^{(1)^T} \mathbf{e}_{i_1}$, $\mathbf{A}^{(2)^T} \mathbf{e}_{i_2}$, and $\mathbf{A}^{(3)^T} \mathbf{e}_{i_3}$ are considered as outputs of three multi-layer perceptrons (MLPs). The inputs of the three branches are $\mathbf{e}_{i_1}$, $\mathbf{e}_{i_2}$, and $\mathbf{e}_{i_3}$, and the outputs are $\bar{\mathbf{a}}_{i_1}^{(1)} = \mathbf{A}^{(1)^T} \mathbf{e}_{i_1}$, $\bar{\mathbf{a}}_{i_2}^{(2)} = \mathbf{A}^{(2)^T} \mathbf{e}_{i_2}$, and $\bar{\mathbf{a}}_{i_3}^{(3)} = \mathbf{A}^{(3)^T} \mathbf{e}_{i_3}$, respectively. Calculating the N-fold contraction of $\bar{\mathbf{a}}_{i_1}^{(1)}$, $\bar{\mathbf{a}}_{i_2}^{(2)}$, and $\bar{\mathbf{a}}_{i_3}^{(3)}$ produces x_{i_1,i_2,i_3}. Identifying with the forward computation of the tri-branch fully-connected network, $\mathbf{A}^{(1)^T}$, $\mathbf{A}^{(2)^T}$ and $\mathbf{A}^{(3)^T}$ can be seen as the weight

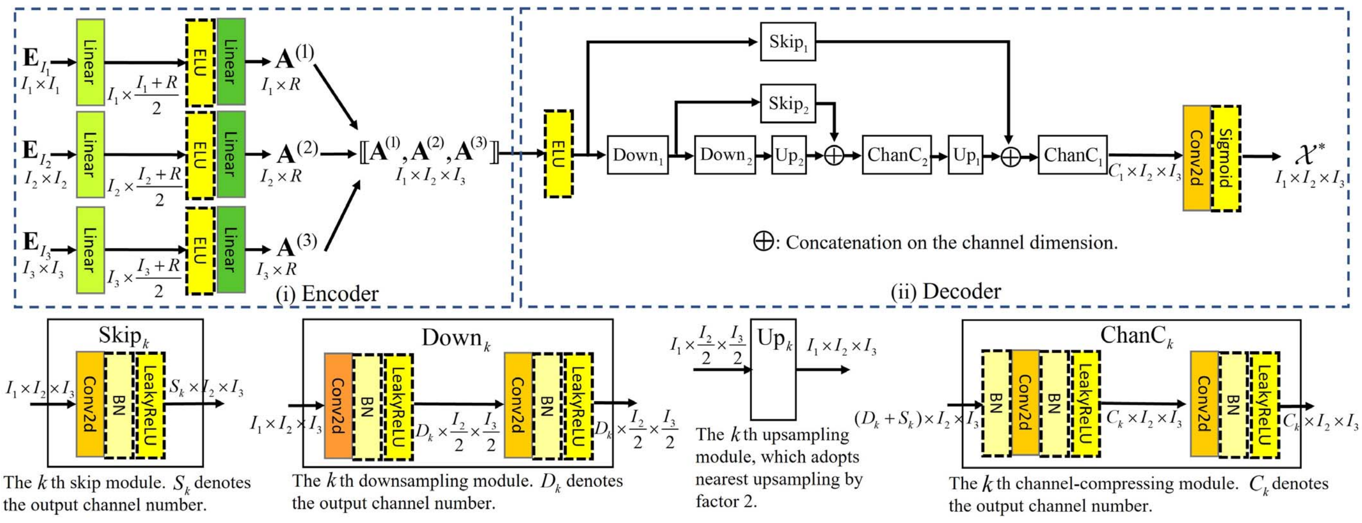
(i) Encoder
(ii) Decoder
Linear
ELU
$\mathbf{E}_{I_1}$ $I_1 \times I_1$
$\mathbf{E}_{I_2}$ $I_2 \times I_2$
$\mathbf{E}_{I_3}$ $I_3 \times I_3$
$I_1 \times \frac{I_1+R}{2}$
$\mathbf{A}^{(1)}$ $I_1 \times R$
$[\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}]\!]$ $I_1 \times I_2 \times I_3$
Skip$_1$
Skip$_2$
Down$_1$
Down$_2$
Up$_2$
Up$_1$
ChanC$_2$
ChanC$_1$
Conv2d
Sigmoid
$\mathcal{X}^*$ $I_1 \times I_2 \times I_3$
⊕: Concatenation on the channel dimension.
BN
LeakyReLU
The kth skip module. S_k denotes the output channel number.
The kth downsampling module. D_k denotes the output channel number.
The kth upsampling module, which adopts nearest upsampling by factor 2.
The kth channel-compressing module. C_k denotes the output channel number.

Fig. 4. Network structure of CNet. The inputs of three branches are three identity matrices, and $[\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}]\!]$ and $\mathcal{X}^*$ are the outputs of encoder and decoder/CNet, respectively. The dimensions of variables in CNet are provided if necessary. The structure of the decoder is described as $[D_1, D_2], [S_1, S_2], [C_1, C_2]$ for simplicity, which represent the channel dimensions in the outputs of the corresponding modules. The dimension changes in each kind of module are provided. The kernel sizes of two convolutional layers in the downsampling module and the first convolutional layer in the channel-compressing module are 3×3. In other convolutional layers, kernel sizes are 1×1. The convolution stride is 2 or 1, which aligns with the dimensions of the output of corresponding convolutional layers. In the experiments, we set $R = 200$ and $D_1 = D_2 = S_1 = S_2 = C_1 = C_2 = 128$.

matrices acquired by aggregating the weights and activation operations in each MLP. Hence, training this network with a least squares cost is equivalent to searching for a minimum of (14).

We use batch input strategy, and thus the identity matrices are adopted as the inputs, where the column vectors of the identity matrices are treated as index vectors. Inputting identity matrices to each branch is equivalent to multiplying the identity matrices and the weight matrices, where the results or network outputs are still weight matrices, e.g., ${\mathbf{A}^{(1)}}^T \mathbf{E}_{I_1} = {\mathbf{A}^{(1)}}^T$ with identity matrix $\mathbf{E}_{I_1} \in \mathbb{R}^{I_1 \times I_1}$. Using identity matrices as inputs of the three branches generates $\mathbf{A}^{(1)}$, $\mathbf{A}^{(2)}$, and $\mathbf{A}^{(3)}$. With these weight matrices, or factor matrices of CP decomposition, we can construct the tensor in a low-rank domain using the Kruskal operation. Then, we rewrite (14) in another equivalent form:

$$\min_{\mathbf{A}^{(1)},\mathbf{A}^{(2)},\mathbf{A}^{(3)}} \left\| \mathfrak{P}_\Omega \left([\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}]\!] \right) - \mathfrak{P}_\Omega (\mathcal{Y}) \right\|_F^2 . \quad (15)$$

Overall, the inputs of the encoder are $\mathbf{E}_{I_1}$, $\mathbf{E}_{I_2}$, $\mathbf{E}_{I_3}$, and the output is $[\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}]\!]$.

The encoder produces an exact low-rank tensor. However, as mentioned before, visual data are not exactly low rank. That is, a low-rank model may not guarantee recovering the detail information. Hence, we introduce a decoder to refine the details of the low-rank visual data. From the rank perspective, the decoder acts as a function that transforms the low-rank tensor to a high-rank domain. Combining with the decoder, we modify (15) as:

$$\min_{\mathbf{A}^{(1)},\mathbf{A}^{(2)},\mathbf{A}^{(3)}} \left\| \mathfrak{P}_\Omega \left(\mathfrak{D} \left([\![\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \mathbf{A}^{(3)}]\!] \right) \right) - \mathfrak{P}_\Omega (\mathcal{Y}) \right\|_F^2 , \quad (16)$$

where $\mathfrak{D}(\cdot)$ denotes the decoder.

We exploit CNN to construct the decoder, which is suitable for image related tasks. Convolution operations involved in CNN can efficiently capture local patterns and features that are present in multiple regions of the image, where the learnable convolution kernels are shared across the entire image. In the past decades, CNN-based algorithms have achieved great success in image restoration [37], [44], where the downsampling and upsampling with skip connection structure is widely used, like U-Net [45]. For example, in [46], the authors employ U-Net for image super resolution, where the input and output are low resolution and high resolution images, respectively. In [47], U-Net is utilized to generate a noise-free image using the input noisy image. Thus, similarly, U-Net is adopted as our decoder for detail enhancement. The input of the decoder is the low-rank approximation, and the output is an image with fine details.

B. Network Structure

The network structure of CNet (16) is illustrated in Fig. 4. For a third-order tensor of dimensions $I_1 \times I_2 \times I_3$, the inputs of the three branches in the encoder are three identity matrices $\mathbf{E}_{I_1} \in \mathbb{R}^{I_1 \times I_1}$, $\mathbf{E}_{I_2} \in \mathbb{R}^{I_2 \times I_2}$, and $\mathbf{E}_{I_3} \in \mathbb{R}^{I_3 \times I_3}$, and the outputs are three factor matrices $\mathbf{A}^{(1)}$, $\mathbf{A}^{(2)}$, and $\mathbf{A}^{(3)}$, respectively. Each branch is composed of two linear layers and one exponential linear unit (ELU) as activation function in between, where the dimension change of matrices is: $I_n \times I_n \to I_n \times (I_n + R)/2 \to I_n \times R$ for $n = 1, 2, 3$. Then, with these factor matrices, a low-rank tensor is constructed by Kruskal operation, which serves as the output of the encoder.

The decoder mainly consists of two downsampling modules, two upsampling modules, two channel-compression modules, and two skip modules. The downsampling operations extract the features of the low-rank visual data, and the upsampling operations combined with channel-compression operations aim to generate visual data with high recovery accuracy from the

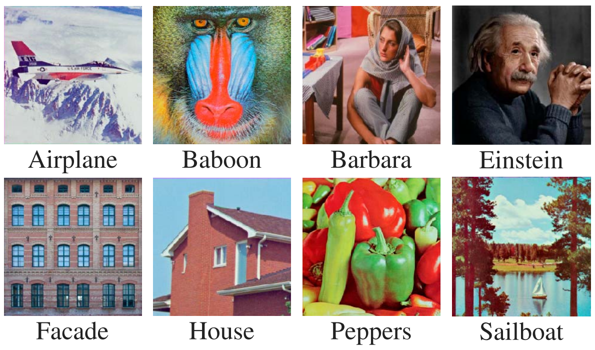

Fig. 5. Eight benchmark color images.

extracted features. The low-rank visual data are first passed through an ELU, as the input of subsequent modules. The downsampling module denoted as Down_k for $k = 1, 2$ has two convolutional layers, each of which is followed by one batch normalization (BN) layer and one LeakyReLU activation function. The kernel sizes and convolution strides of the two convolutional layers in Down_k are 3×3, 2 and 3×3, 1, respectively. The skip module Skip_k is composed of one convolutional layer, one BN layer, and one LeakyReLU function, where the kernel sizes and convolution strides are 1×1 and 1, respectively. The upsampling module Up_k upsamples the outputs of downsampling module Down_k or previous channel-compression module by factor 2, and then the concatenation of the upsampled tensor and the output of Skip_k on the channel dimension act as the input of channel-compression module ChanC_k. The composition of ChanC_k is similar to Down_k, except for one more BN layer at the beginning. The two convolutional layers in ChanC_k have kernel sizes of 3×3 and 1×1, respectively, where the convolution strides are both 1. The output of the final channel-compression module undergoes an additional step to convert it into an $I_1 \times I_2 \times I_3$ tensor, which is achieved by passing the output through a convolutional layer and applying a sigmoid activation function, resulting in the final output of the CNet. Note that the channel numbers in the output of Down_k, Skip_k, and ChanC_k are D_k, S_k, and C_k, respectively. For ease of representation, the structure of the decoder in Fig. 4 is denoted as $[D_1, D_2]$, $[S_1, S_2]$, $[C_1, C_2]$. Ablation studies about the encoder and decoder structures will be detailed in Section IV.

Differences with DIP: In terms of the network structure, the decoder is similar to the DIP. However, they serve different purposes: one refines a coarse estimate, the other imputes missing entries. Specifically, the former aims to refine the coarse visual data generated by the encoder, while the latter is designed to complete the missing entries. For CNet, the completion task is implemented using the encoder, devised based on CP decomposition. Compared with prevailing neural networks including DIP that are referred to as "black box", the CNet is designed based on the low rank TC model, explaining why the missing elements can be recovered.

For a partially observed tensor $\mathcal{Y}$ under index set Ω, given its observed elements, we construct the index vectors $\mathbf{e}_{i_1}$, $\mathbf{e}_{i_2}$, and $\mathbf{e}_{i_3}$. These index vectors combined with the observed elements are considered as the training data. The inputs of CNet are $\mathbf{E}_{I_1}$, $\mathbf{E}_{I_2}$, and $\mathbf{E}_{I_3}$, whose column vectors are index vectors. The difference between the outcome of CNet and $\mathcal{Y}$ on the index set Ω is used to calculate the loss.

Algorithm 1 CNet for third-order tensor completion

```
Input: Partially-observed tensor $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ with index
  set $\Omega$, identity matrices $\mathbf{E}_{I_1} \in \mathbb{R}^{I_1 \times I_1}$, $\mathbf{E}_{I_2} \in \mathbb{R}^{I_2 \times I_2}$, and
  $\mathbf{E}_{I_3} \in \mathbb{R}^{I_3 \times I_3}$, learning rate $lr = 0.002$, iteration number
  $Iter = 10000$, CP decomposition rank $R = 200$, $\varepsilon = 0.001$,
  network parameters $\Theta^0$ initialized by Pytorch default initialization method.
  1. Training
  for $iter = 0$ to $Iter$ do
      Update $\Theta^{iter}$ by minimizing (17) using Adam optimizer.
  end for
  $\Theta^* = \Theta^{Iter}$.
  2. Testing
  Calculate $\mathcal{X}^* = \mathfrak{F}(\mathbf{E}_{I_1}, \mathbf{E}_{I_2}, \mathbf{E}_{I_3} | \Theta^*)$.
Output: Complete tensor $\mathcal{X}^*$.
```

Rather than the commonly used mean squared error (MSE) loss, we select the Charbonnier loss [48] for CNet. Since we employ the observed incomplete tensor to train the network, where both the encoder and decoder are not pre-trained, the prior information is so limited. If not properly trained, CNet may generate artefacts (seen as outliers). Therefore, we prefer a robust measure, namely, the Charbonnier loss:

$$\mathrm{Char}_{\mathrm{loss}}(y, \hat{y}) = \sqrt{(\hat{y} - y)^2 + \varepsilon^2}.$$

Here, y is the label, $\hat{y}$ is the network output, and ε is a constant and set as (10^{-3}) in the experiments. Introducing a small constant ε makes the Charbonnier loss differentiable at the zero point.

Then, the training process solves the following minimization problem:

$$\Theta^* = \arg\min_{\Theta} \times \sqrt{\left\| \mathfrak{P}_\Omega \left(\mathfrak{F}(\mathbf{E}_{I_1}, \mathbf{E}_{I_2}, \mathbf{E}_{I_3} | \Theta) \right) - \mathfrak{P}_\Omega (\mathcal{Y}) \right\|_F^2 + \varepsilon^2}, \tag{17}$$

where $\mathfrak{F}(\cdot|\Theta)$ denotes the network function with learnable parameter set Θ. With the trained CNet, the recovered tensor is computed as:

$$\mathcal{X}^* = \mathfrak{F}(\mathbf{E}_{I_1}, \mathbf{E}_{I_2}, \mathbf{E}_{I_3} | \Theta^*). \tag{18}$$

We summarize our method in Algorithm 1.

IV. Ablation Studies

In this section, we investigate CNet from different aspects, including hyperparameter setting, network structure, and its mechanism. The test data include color images, HSIs, and color videos, which will be introduced in Section V. For a test tensor, a specified proportion of entries are randomly selected as observed elements, which is referred to as the observation ratio

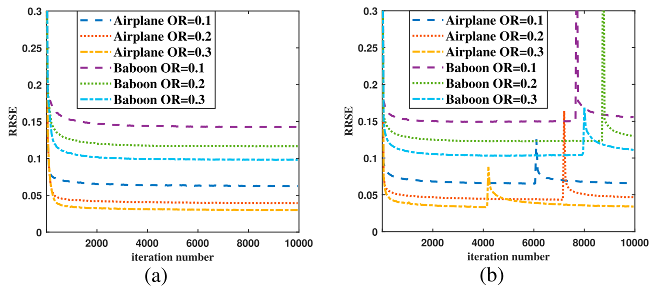

Fig. 6. RRSEs of restored images *Airplane* and *Baboon* using the proposed method (a) and DIP (b) during training at OR = 0.1, 0.2, 0.3. For both (a) and (b), the learning rate is 0.002.

(OR). The definitions of the evaluation metrics, viz. relative root squared error (RRSE) and peak signal-to-noise ratio (PSNR), can also be found in Section V.

A. Training

To train CNet, we utilize the Adam optimizer, and the network parameters are initialized by default schemes in Pytorch. The learning rate is set as 0.002. To determine the number of training iterations, we show the RRSEs of CNet output during training for images *Airplane* and *Baboon* in Fig. 6(a), where different ORs are tried. Apparently, the RRSE decreases as the training progresses. Specifically, for 10000 iterations, there are no overfitting or RRSE spikes. Large learning rates are also examined, like 0.01, and the conclusion remains the same. Thus, we fix the iteration number as 10000 in all the experiments.

Moreover, we explore the training of DIP for the same situation. Since the height and width of these color images are both 256, the input of DIP is a tensor of dimensions 256×256×32 filled by uniform noise [39]. According to the authors' suggestion, the input should be perturbed randomly at every iteration to acquire better results. We find the training is not always stable, namely, gradient spikes and loss spikes exist, resulting in RRSE spikes during training. The unstable training processes are plotted in Fig. 6(b). We further observe that the RRSEs are slightly larger than those before the spikes, viz., recovery performance degrades after spikes. We have also tried to avoid perturbing the input of DIP during training. Nonetheless, the RRSE spikes still occur, and its performance degrades a lot. We may need to set a small learning rate for DIP. However, small learning rates result in longer training time. By contrast, the training of CNet is stable.

B. Network Structure

As for the encoder design, the major concern is to determine the number of linear layers in each branch of MLP. Thus, we conduct experiments for its determination. For ease of representation, we denote the nth layer of MLP as $f_n(\cdot)$. When the input of MLP is $\mathbf{x} \in \mathbb{R}^I$, the final output is $f(\mathbf{x}) \in \mathbb{R}^R$. For simplicity, for an N-layer perceptron, we adopt the dimensions of each layer output as an arithmetic sequence. That is, $f_n(\mathbf{x}) \in \mathbb{R}^{I-\frac{n}{N}(I-R)}$. Then, we use color images *Airplane*, *Barbara*, *Facade*, and *Sailboat* to evaluate the performance of different

TABLE I
PSNRs of Recovered Color Images *Airplane*, *Barbara*, *Facade*, and *Sailboat* for Different Encoder Structures and ORs. The Activation Function Is ELU, and The Decoder Structure Is [128, 128], [128, 128], [128, 128]

Image	OR	Layer number of each branch MLP			
		1	2	3	4
Airplane	0.1	26.95	27.21	26.91	26.54
	0.5	37.79	37.93	37.50	37.18
	0.8	44.68	44.82	44.47	44.22
Barbara	0.1	26.88	27.36	26.52	26.05
	0.5	38.90	39.21	37.95	36.92
	0.8	47.23	47.39	46.84	46.19
Facade	0.1	28.13	28.50	28.23	27.94
	0.5	37.85	37.98	37.48	37.10
	0.8	44.62	44.70	44.36	44.07
Sailboat	0.1	24.22	24.40	24.18	23.94
	0.5	32.78	32.98	32.66	32.40
	0.8	38.55	38.83	38.44	38.24

encoder structures under various ORs, where different numbers of layers in each branch of MLPs are tested. The results are tabulated in Table I, and we find more layers do not result in improved performance, while two-layer perceptron performs better than other structures. Therefore, we choose the two-layer structure. Besides, we fix the number of linear layers as two and investigate the impact of different activation functions. The results are shown in the supplementary material. We see that ELU outperforms the remaining choices, and hence it is selected as the activation function.

For the decoder, the results of different structures are reported in the supplementary material. In particular, to illustrate how the decoder depth changes, the structures for $[D_1]$, $[S_1]$, $[C_1]$, and $[D_1, D_2, D_3]$, $[S_1, S_2, S_3]$, $[C_1, C_2, C_3]$ are also included. The structure for $[D_1, D_2]$, $[S_1, S_2]$, $[C_1, C_2]$ is depicted in Fig. 4. From the results, we see that the deeper the decoder, the better the performance. However, when the decoder is deep enough, a deeper depth will not cause further performance improvement. To balance the training time and recovery performance, we prefer the decoder to have fewer learnable parameters. Therefore, in the experiments, we choose [128, 128], [128, 128], [128, 128] for the decoder structure.

C. Rank

Another parameter to be determined is the rank R. We vary the value of R from 20 to 260 to recover images *Airplane*, *Peppers*, and *Sailboat* when OR is 0.8. The dimensions of these images are 256×256×3. The PSNRs of recovered images versus rank are plotted in Fig. 7. We observe that a larger rank produces a higher PSNR. Besides, the recovery performance is not sensitive to R. Generally, a large rank would increase the number of network parameters, and some visual data contain intricate details that result in a relatively large rank. Therefore, balancing these two aspects, R is fixed as 200 for all other experiments. Note that for a third-order $I_1 \times I_2 \times I_3$ tensor, its CP rank is bounded above by $\min(I_1 I_2, I_2 I_3, I_1 I_3)$ [11]. Hence, it is reasonable to set the CP rank of CNet as 200.

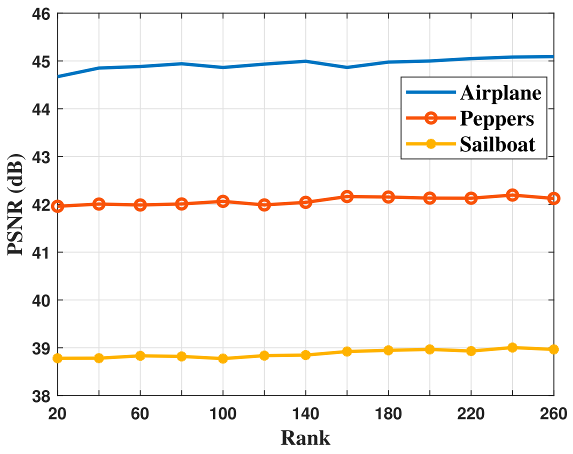

Fig. 7. Rank versus PSNR for restored *Airplane*, *Peppers*, and *Sailboat* when OR is 0.8. We set rank as 200 in all the other experiments.

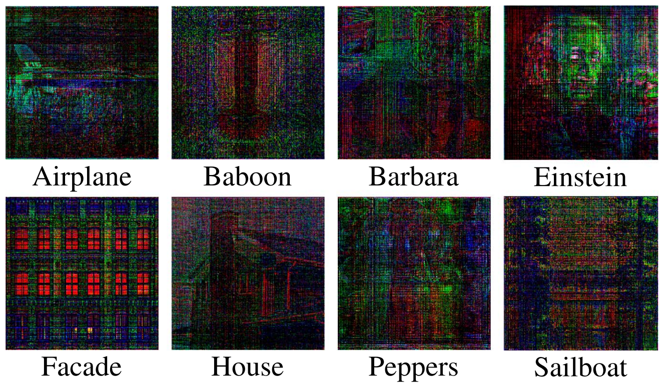

Fig. 8. Encoder outputs for benchmark color images in Fig. 5 when OR is 1.

TABLE II
PSNRs of Recovered Hyperspectral Images *PaviaU*, *Urban*, and *WDC_mall*, and Color Videos *Bridge_close*, *Bus*, and *News* for Different Ranks When OR Is 0.8

Data	Rank		
	20	200	300
PaviaU (340×340×50)	38.87	39.70	39.97
Urban (307×307×50)	39.18	39.92	40.09
WDC_mall (307×307×50)	43.35	44.43	44.97
Bridge_close (144×176×3×30)	42.14	43.28	43.30
Bus (144×176×3×30)	35.32	37.35	37.98
News (144×176×3×30)	46.32	47.25	47.34

TABLE III
PSNRs of Recovered Color Image *Airplane*, *Barbara*, *Facade*, and *Sailboat* for Charbonnier Loss and MSE Loss

Image	OR	Loss function	
		MSE loss	Charbonnier loss
Airplane	0.1	27.09	27.21
	0.8	44.50	44.82
Barbara	0.1	27.10	27.36
	0.8	46.56	47.39
Facade	0.1	28.37	28.50
	0.8	44.20	44.70
Sailboat	0.1	24.26	24.40
	0.8	38.64	38.83

To further validate our choice of R, we utilize HSIs and color videos to conduct TC when OR is 0.8. The rank is set as 20, 200, and 300. The PSNRs of recovery results are tabulated in Table II. As we can see, for most cases, the differences between $R = 200$ and $R = 20$ are around 1 dB. For color video *Bus*, the PSNR gap even achieves 2 dB. Comparing with $R = 200$ and $R = 300$, the PSNR differences are not big. For *Urban* and *Bridge_close*, the PSNRs for $R = 200$ and $R = 300$ are nearly the same. In summary, $R = 200$ is a reasonable choice.

D. Loss Function

Comparing with the MSE loss, it has been demonstrated that the Charbonnier loss is more robust to outliers and can preserve sharp details on image restoration tasks [49]. To further validate the effectiveness of the Charbonnier loss, we train CNet using Charbonnier loss and MSE loss on color image inpaiting task, while all other settings remain unchanged. The results are tabulated in Table III. We see that the Charbonnier loss performs slightly better than the MSE loss. It is worth pointing out that the excellent performance of CNet mainly comes from the network design, rather than the choice of loss function. Both MSE loss and Charbonnier loss perform well, and the latter performs a little better.

TABLE IV
PSNRs of Recovery Results of Eight Benchmark Color Images With Encoder Alone, DIP (Skip-2), and CNet (Encoder Plus Decoder) at OR = 0.8

Image	Encoder	DIP (Skip-2)	CNet
Airplane	34.25	43.49	44.82
Baboon	29.32	34.21	35.04
Barbara	34.33	44.24	47.39
Einstein	38.39	49.39	51.77
Facade	36.30	42.81	44.70
House	33.83	42.16	43.17
Peppers	31.75	41.00	42.03
Sailboat	30.73	37.98	38.83

TABLE V
PSNRs of Recovered Color Videos *Bridge_close*, *Bus*, and *News* With Encoder Alone, DIP (Skip-2), and CNet (Encoder Plus Decoder) at OR = 0.8

Image	Encoder	DIP (Skip-2)	CNet
Bridge_close	31.75	35.58	43.28
Bus	21.90	31.42	37.35
News	32.85	40.67	47.25

E. Effectiveness of Encoder and Decoder

To demonstrate the effectiveness of the encoder in constructing low-rank approximations of the test data, we display the encoder outputs for eight benchmark color images in Fig. 8 when CNet is trained for image inpainting. It is observed that these low-rank tensors contain much edge information and can be treated as the low-rank approximation of the corresponding color images. Due to the joint training of the encoder and

TABLE VI
COMPLETION RESULTS OF COLOR IMAGE *FACADE* UNDER DIFFERENT ORS. RRSES, PSNRS, SSIMS, AND RUNTIME ARE LISTED. THE BEST VALUES ARE IN BOLD FACE. THE RESULTS OF COLOR IMAGES *AIRPLANE*, *BABOON*, *BARBARA*, AND *SAILBOAT* ARE LISTED IN THE SUPPLEMENTARY MATERIAL

Image	OR	0.1				0.2				0.3			
	Method	RRSE	PSNR (dB)	SSIM	Time	RRSE	PSNR (dB)	SSIM	Time	RRSE	PSNR (dB)	SSIM	Time
Facade	CoNoT	0.1133	23.53	0.8422	166.87	0.0856	25.97	0.9060	236.22	0.0686	27.89	0.9363	427.37
	FBCP	0.1053	25.54	0.8136	35.46	0.0785	28.32	0.8886	40.71	0.0685	29.99	0.9220	42.27
	FBCP-MP	0.1017	25.85	0.8208	52.75	0.0836	27.86	0.8832	60.84	0.0734	29.46	0.9199	69.61
	HaLRTC	0.1108	24.86	0.8013	4.30	0.0814	27.53	0.8808	3.98	0.0639	29.63	0.9213	3.83
	HLRTF	0.0932	26.36	0.8385	4.52	0.0622	29.87	0.9209	5.44	0.0480	32.12	0.9506	5.17
	LRMF	0.1283	23.58	0.7420	5.21	0.0968	26.03	0.8328	4.88	0.0793	27.76	0.8779	4.72
	LRTF	0.0979	25.93	0.8188	12.09	0.0721	28.59	0.8931	11.71	0.0580	30.47	0.9264	11.96
	DIP	0.0965	26.06	0.8404	254.35	0.0649	29.51	0.9211	254.51	0.0481	32.11	0.9540	254.11
	t-LogDet	0.1004	25.71	0.8172	2.29	0.0699	28.86	0.9011	2.29	0.0531	31.25	0.9395	2.24
	t-ϵ-LogDet	0.0969	26.02	0.8214	**1.78**	0.0693	28.93	0.9003	**1.79**	0.0549	30.95	0.9357	**1.74**
	Proposed	**0.0729**	**28.50**	**0.8970**	216.05	**0.0507**	**31.65**	**0.9471**	215.82	**0.0392**	**33.87**	**0.9669**	215.19

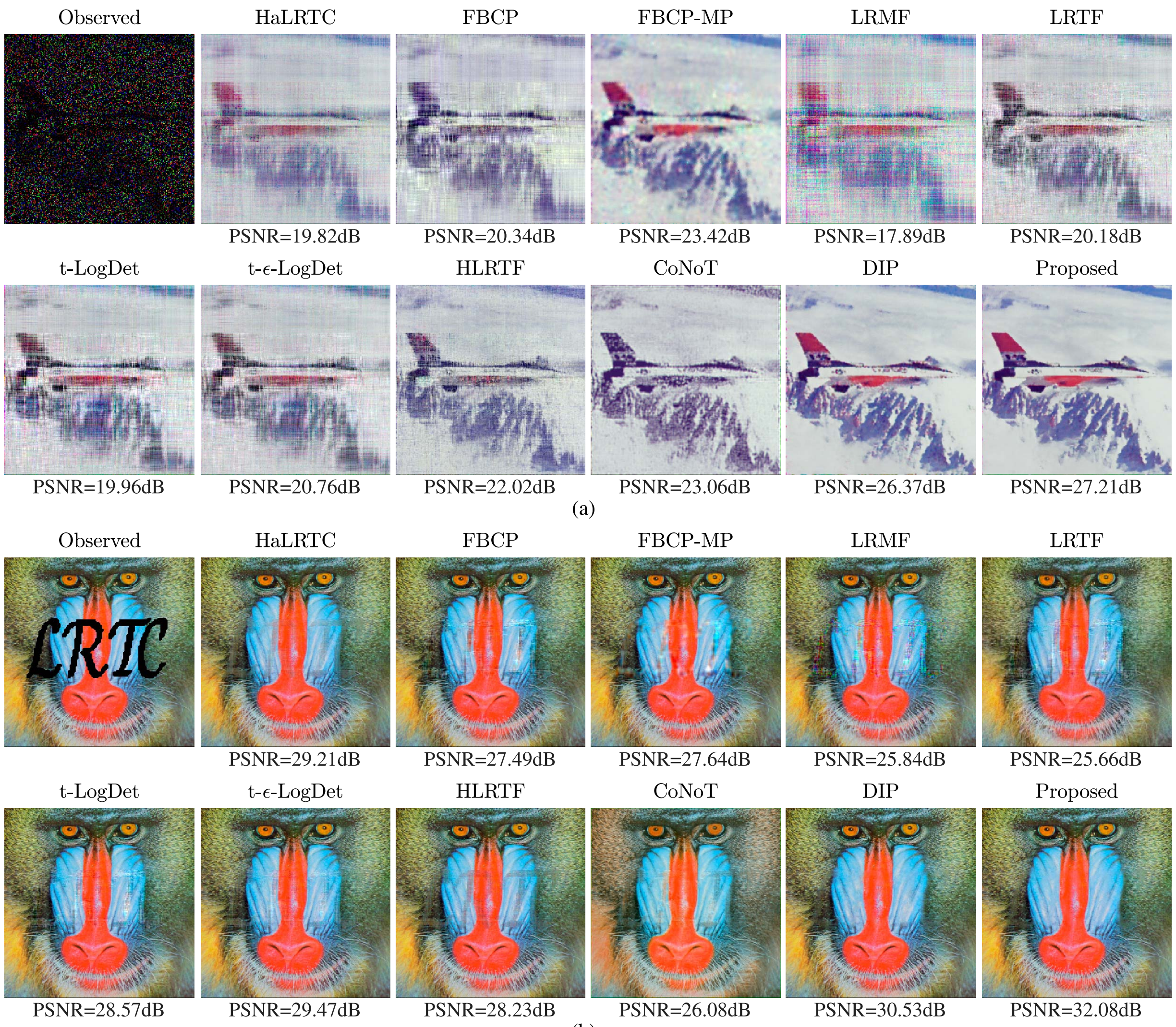

Fig. 9. (a) Recovered *Airplane* by different algorithms at OR = 0.1. (b) Recovered *Baboon* with text masks by different algorithms. For both (a) and (b), top to bottom, left to right correspond to original data, observed data, recovered results by HaLRTC, FBCP, FBCP-MP, LRMF, LRTF, t-LogDet, t-ϵ-LogDet, HLRTF, CoNoT, DIP, and the proposed method, respectively.

decoder, the encoder captures edge-type information in a low-rank representation, and then the decoder fills the smooth areas. If the encoder is trained alone for image inpainiting, we will get low-rank approximations of the original color images, and smooth areas will be approximated by the CP decomposition itself.

Besides, we conduct experiments to investigate the effectiveness of the decoder and encoder separately. The results are tabulated in Tables IV and V. Note that only the decoder itself cannot perform TC. The results by decoder are acquired by setting the input of decoder as random noise, which utilizes the mechanism of DIP. Thus, we state the results are obtained

TABLE VII
INPAINTING RESULTS OF COLOR IMAGE *HOUSE* UNDER TEXT MASKS. RRSES, PSNRS, AND SSIMS ARE LISTED. THE BEST VALUES ARE IN BOLD FACE. THE RESULTS OF COLOR IMAGES *BABOON*, *EINSTEIN*, AND *PEPPERS* ARE TABULATED IN THE SUPPLEMENTARY MATERIAL

Image	Method	RRSE	PSNR (dB)	SSIM
House	CoNoT	0.0614	28.89	0.9646
	FBCP	0.0695	27.81	0.9659
	FBCP-MP	0.0772	27.14	0.9622
	HaLRTC	0.0633	28.59	0.9736
	HLRTF	0.0796	26.59	0.9637
	LRMF	0.0664	28.17	0.9639
	LRTF	0.0643	28.44	0.9658
	DIP	0.0388	32.83	0.9817
	t-LogDet	0.0669	28.10	0.9719
	t-ϵ-LogDet	0.0638	28.51	0.9733
	Proposed	**0.0303**	**34.98**	**0.9908**

by "DIP (Skip-2)" instead of "decoder". For color images, the CNet achieves around 1 to 3 dB higher PSNR than the DIP (Skip-2), while the superiority is remarkable (more than 6 dB) for video data. The main reasons for such difference might be as follows. For a color image with three channels, the contents of each channel are very similar, indicating that the data structure is simple. Therefore, both the DIP (Skip-2) and CNet perform well. In other words, both the low-rank prior and the implicit image prior captured by network parametrization work well. However, for a video, the contents of consecutive frames are continuously changing, and thus its structure is more complex than the color image. The experimental results exhibit that the **low-rank prior is more effective at capturing the intricate correlations within the data compared to the implicit image prior**. On the other hand, comparing the results by the encoder and CNet, it is clear that the decoder significantly improves the recovery performance, validating our motivation of handling visual data completion from the perspective of approximate low-rankness.

V. EXPERIMENTAL RESULTS

In this section, we evaluate the performance of CNet for the inpainting task on color images, grayscale videos, HSIs, color videos, and light field images.

We compare the CNet with state-of-the-art TC algorithms, including FBCP [17], FBCP-MP [17], HaLRTC [7], HLRTF [50], CoNoT [51], LRMF [52], LRTF [52], t-LogDet [53], t-ϵ-LogDet [53] as well as DIP [39]. HaLRTC minimizes the tensor multilinear ranks to perform TC, while LRMF, LRTF, t-LogDet, and t-ϵ-LogDet minimize the tensor tubal rank using different rank surrogates. HLRTF handles TC via low-rank tensor factorization based on t-SVD, in which a neural network is designed. CoNoT is a tensor nuclear norm based method, where CNN are utilized to perform spatial and spectral/temporal transforms on tensors. FBCP and FBCP-MP develop a TC algorithm utilizing deterministic Bayesian inference based on CP decomposition. For DIP, we adopt its representative implementation, which consists of five downsamplings and five nearest neighbor upsamplings with skip connections (Skip-5). For a color image of dimensions 256×256×3, the numbers of learnable parameters for CNet, Skip-5, and HLRTF are 1,320,031, 3,002,627, and 784,350, respectively.

Regarding parameter setting, for HLRTF, CoNoT, and DIP, the number of iterations is treated as a hyperparameter. For an incomplete visual data, we train the network with different iteration numbers and then select the one that results in the best performance. Besides, the maximum number of iterations is 10000. The hyperparameters of other algorithms are chosen according to the values suggested in the corresponding works. The CNet, HLRTF, CoNoT, and DIP are programmed using Pytorch 1.11.0 framework, and experiments are implemented on a computer with 1.8 GHz 16-core CPU and eight NVIDIA RTX 2080Ti 11GB GPUs. Other competing methods are run in MATLAB R2021a on a computer with 2.9GHz CPU and 16 GB memory.

The evaluation metrics for the recovery performance include RRSE, PSNR, and structural similarity index (SSIM) [54]. RRSE is defined as:

$$\mathrm{RRSE} = \frac{\|\mathcal{X}^* - \mathcal{X}\|_F}{\|\mathcal{X}\|_F}, \tag{19}$$

where $\mathcal{X}^*$ and $\mathcal{X}$ are the recovered tensor and the ground truth, respectively. PSNR is computed as:

$$\mathrm{PSNR} = 10\log_{10}\frac{(\max(\mathcal{X}))^2}{\|\mathcal{X}^* - \mathcal{X}\|_F^2 / \prod_{n=1}^{N} I_n},$$

where $\max(\mathcal{X})$ returns the maximum element of $\mathcal{X}$. SSIM is calculated as:

$$\mathrm{SSIM} = \frac{(2\mu_{\mathcal{X}}\mu_{\mathcal{X}^*} + B_1)(2\sigma_{\mathcal{X}\mathcal{X}^*} + B_2)}{(\mu_{\mathcal{X}}^2 + \mu_{\mathcal{X}^*}^2 + B_1)(\sigma_{\mathcal{X}}^2 + \sigma_{\mathcal{X}^*}^2 + B_2)},$$

where $\mu_{\mathcal{X}}$, $\sigma_{\mathcal{X}}$, and $\mu_{\mathcal{X}^*}$, $\sigma_{\mathcal{X}^*}$ denote the mean and standard deviation of the elements of $\mathcal{X}$ and $\mathcal{X}^*$, respectively. Similarly, $\sigma_{\mathcal{X}\mathcal{X}^*}$ represents the cross covariance of $\mathcal{X}$ and $\mathcal{X}^*$ elements. In the experiments, we use the MATLAB built-in command $\mathrm{ssim}()$ to compute SSIM. By default, $B_1 = (0.01L)^2$ and $B_2 = (0.03L)^2$, where L is the dynamic range of tensor element values.

A. *Color Images*

We compare the performance of different algorithms on color image inpainting using benchmark images shown in Fig. 5. These images are of dimensions 256×256×3 and can be modeled as third-order tensors.

First, we set OR as 0.1, 0.2, and 0.3 for *Airplane*, *Baboon*, *Barbara*, *Facade*, and *Sailboat*. The restored results of these images by different algorithms are reported in Table VI. The best values are highlighted in bold face. We find that CNet performs well in terms of all the metrics. In addition, the runtime of different algorithms is provided. Compared with DIP, CNet exhibits better performance with fewer parameters and shorter runtime in most cases. The shorter runtime of DIP means that the number of iterations is smaller than 10000, which is determined by the aforementioned strategy. It is worth pointing out that the CNet does not require hyperparameter adjustments. For other algorithms, they need to tune certain hyperparameters, where the time consumed cannot be neglected. Besides, in practical applications, the iteration number of CNet can be reduced to yield shorter runtime. According to our experience, CNet is able to produce a good recovery

TABLE VIII

Recovery Results of Grayscale Video *Akiyo* and HSI *PaviaU* Under Different ORs. RRSEs, PSNRs, and SSIMs Are Listed. The Best Values Are in Bold Face. The Results of Grayscale Videos *Claire* and *Highway*, and HSIs *Urban* and *WDC_mall* Are Listed in the Supplementary Material

Grayscale Video/HSI	OR	0.05			0.1			0.15		
	Method	RRSE	PSNR (dB)	SSIM	RRSE	PSNR (dB)	SSIM	RRSE	PSNR (dB)	SSIM
Akiyo (144×176×50)	CoNoT	0.0625	31.78	0.9454	0.0468	34.30	0.9661	0.0382	36.06	0.9767
	FBCP	0.0943	28.21	0.8221	0.0677	31.09	0.8993	0.0557	32.77	0.9313
	FBCP-MP	0.1215	26.01	0.7142	0.1022	27.51	0.7969	0.0943	28.21	0.8264
	HaLRTC	0.1925	22.01	0.6662	0.1328	25.23	0.7795	0.1048	27.29	0.8442
	HLRTF	0.0639	31.60	0.9351	0.0430	35.03	0.9717	0.0329	37.37	0.9833
	LRTF	0.0771	29.95	0.8760	0.0573	32.53	0.9327	0.0492	33.86	0.9486
	Deep_HS	0.0546	32.96	0.9577	0.0375	36.22	0.9781	0.0282	38.70	0.9862
	t-LogDet	0.0724	30.50	0.8944	0.0535	33.14	0.9416	0.0430	35.04	0.9619
	t-ϵ-LogDet	0.0705	30.74	0.9019	0.0491	33.89	0.9535	0.0391	35.86	0.9707
	Proposed	**0.0479**	**34.09**	**0.9662**	**0.0323**	**37.51**	**0.9847**	**0.0255**	**39.57**	**0.9904**
PaviaU (340×340×50)	CoNoT	0.0814	28.31	0.9369	0.0512	32.34	0.9715	0.0385	34.82	0.9826
	FBCP	0.2693	17.93	0.4839	0.1975	20.63	0.6581	0.1691	21.99	0.7279
	FBCP-MP	0.2412	18.88	0.5267	0.2142	19.91	0.6113	0.2023	20.41	0.6461
	HaLRTC	0.3052	16.83	0.5959	0.1304	24.22	0.8663	0.0699	29.62	0.9499
	HLRTF	0.1152	25.30	0.8632	0.0596	31.02	0.9539	0.0409	34.30	0.9770
	LRTF	0.2688	17.98	0.5294	0.2044	20.38	0.6726	0.1681	22.07	0.7511
	Deep_HS	0.0847	27.97	0.9313	0.0438	33.70	0.9765	0.0330	36.15	0.9843
	t-LogDet	0.2451	18.73	0.5926	0.1716	21.83	0.7417	0.1376	23.75	0.8061
	t-ϵ-LogDet	0.3153	16.55	0.4258	0.1522	22.88	0.7867	0.1151	25.30	0.8595
	Proposed	**0.0730**	**29.26**	**0.9462**	**0.0413**	**34.20**	**0.9780**	**0.0309**	**36.73**	**0.9859**

result within 4000 iterations. We also depict the recovered *Airplane* when OR is 0.1 in Fig. 9(a). It is seen that the results restored by HaLRTC, LRMF, LRTF, t-LogDet, t-ϵ-LogDet, and HLRTF are blurred and contain many artifacts. The outcome by FBCP-MP is slightly opaque and has pseudo colors. In contrast, CNet is able to recover more vivid details and structural information, resulting in a recovered image that is highly similar to the original one. Besides, we choose *House*, *Einstein*, *Peppers* and randomly select $10\% - 50\%$ of the pixels to test different methods. The results are tabulated in the supplementary material, demonstrating that the CNet outperforms other algorithms in terms of RRSE, PSNR, and SSIM for all the selected ORs.

Next, we investigate the performance of different methods in the presence of text masks. The inpainting results of *Baboon*, *Einstein*, *House*, and *Peppers* are listed in Table VII. It is evident that CNet achieves better performance than its competitors in all metrics. Compared with DIP which performs the second best, CNet achieves better results with an improvement of approximately 1 to 2 dB. The recovered *Baboon* is displayed in Fig. 9(b), and we see that the CNet restores the image more naturally.

The above experimental results demonstrate the outstanding performance of CNet. Although CNet, FBCP-MP, and FBCP all employ CP decomposition, the neural network-based CNet is more effective than those based on the Bayesian inference.

B. Grayscale Videos & Hyperspectral Images

Due to strong correlations among successive frames, grayscale videos have much redundant information. Therefore, grayscale videos have the low-rank structure [26]. In the experiments, we adopt *Akiyo*, *Claire*, and *Highway*[3] in the QCIF format to compare different algorithms. Instead of DIP, we compare CNet with deep hyperspectral prior (Deep_HS) [55], which is more suitable for higher dimensional data. Each video sequence has at least 300 frames, and we just choose the first 50 continuous frames for assessment, namely, each video used in the experiments is represented as a tensor with dimensions of 144×176×50. We randomly sample these three videos with three OR levels, viz., 0.05, 0.1, and 0.15. The RRSEs, PSNRs, and SSIMs of the recovered videos are tabulated in Table VIII. We also illustrate the fifth frame of the restored *Highway* in Fig. 10(a). From these results, we conclude that the CNet performs better and recovers video frames with more details.

To further exhibit the effectiveness of CNet on visual data, we use HSI *University of Pavia* (*PaviaU*), *Urban*, and *Washington DC hall* (*WDC_ hall*)[4], to compare CNet with HaLRTC, FBCP, FBCP-MP, LRTF, t-LogDet, t-ϵ-LogDet, HLRTF, CoNoT, and Deep_HS. Each of the selected HSIs has more than 200 wavebands, and we select 50 successive bands. ORs are chosen as 0.05, 0.1, and 0.15. The recovery results of *PaviaU*, *Urban*, and *WDC_mall* are tabulated in Table VIII. As we can see, CNet still outperforms the remaining methods. In addition, we show one recovery band of *Urban* in Fig. 10(b). The results recovered by CNet contain more detail information.

C. Color Videos

By adding one more MLP branch into the encoder with an identity matrix input, the quad-branch MLPs produce four factor matrices. Utilizing the Kruskal operator, the output of the encoder becomes a fourth-order tensor, which is further processed by the decoder to generate the final output. Thus, with one more MLP branch, CNet can be extended for fourth-order visual data recovery. The encoder structure for fourth-order TC is shown in the supplementary material. In this subsection, we compare the extended CNet with HaLRTC, FBCP, FBCP-MP, DIP, HTNN-FFT, and HTNN-DCT [41] on color video restoration. HTNN-FFT and HTNN-DCT are

[3]http://trace.eas.asu.edu/yuv

[4]http://lesun.weebly.com/hyperspectral-data-set.html

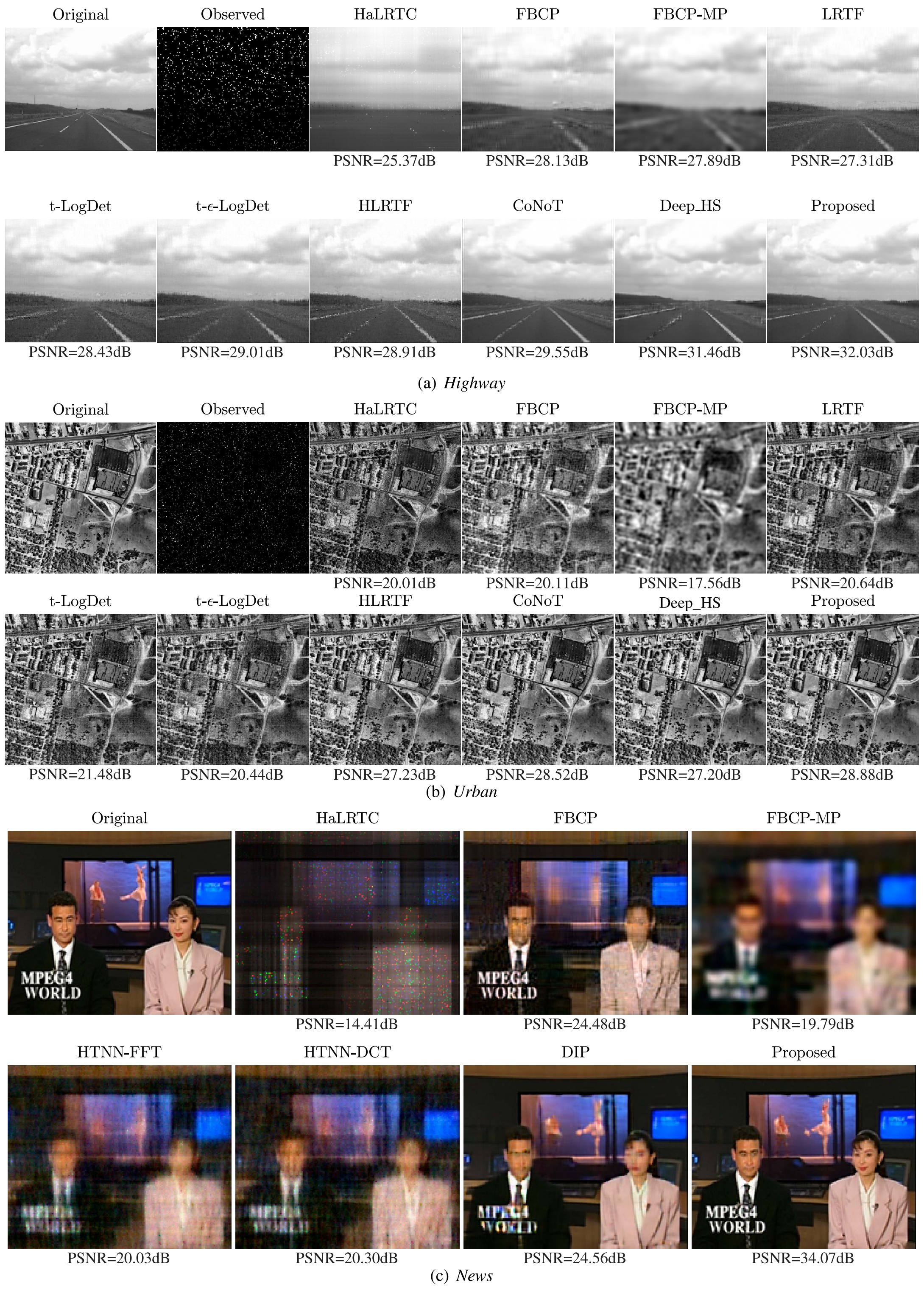

Fig. 10. (a) The 5th frame of inpainting results of grayscale video *Highway* at OR = 0.05. (b) The first band of completion result of HSI *Urban* at OR = 0.05. (c) The 5th frame of recovery results of color video *News* at OR = 0.05. PSNR is calculated based on the whole video sequence or HSI image.

proposed for high-order TC problems, which generalize the definitions of t-SVD and tensor tubal rank to higher order cases. Color videos *Bridge_close*, *Bus*, and *News*[5] in the QCIF format are employed to evaluate their performance. For each video, we select the first successive 30 frames, and thus the dimensions of each color video are $144\times176\times3\times30$. The recovery results at OR = 0.05, 0.1, 0.15 are tabulated in Table IX, where the SSIMs correspond to the average of all the frames. We see that

[5]https://media.xiph.org/video/derf/

TABLE IX
COMPLETION RESULTS OF COLOR VIDEO *BRIDGE_CLOSE* UNDER DIFFERENT ORS. RRSES, PSNRS, AND SSIMS ARE LISTED. THE SSIM IS BASED ON THE AVERAGE OF ALL THE FRAMES. THE BEST VALUES ARE IN BOLD FACE. THE RESULTS OF COLOR VIDEOS *NEWS* AND *BUS* ARE LISTED IN THE SUPPLEMENTARY MATERIAL

Color Video	OR	0.05			0.1			0.15		
	Method	RRSE	PSNR (dB)	SSIM	RRSE	PSNR (dB)	SSIM	RRSE	PSNR (dB)	SSIM
Bridge_close (144×176×3×30)	FBCP	0.0690	25.93	0.7800	0.0553	27.86	0.8379	0.0473	29.21	0.8729
	FBCP-MP	0.0929	23.32	0.6969	0.0882	23.76	0.7208	0.0802	24.59	0.7558
	HaLRTC	0.1451	19.41	0.5471	0.0983	22.80	0.6952	0.0799	24.60	0.7744
	HTNN-FFT	0.0938	23.21	0.6806	0.0726	25.43	0.7809	0.0623	26.76	0.8333
	HTNN-DCT	0.0887	23.69	0.7017	0.0688	25.89	0.7948	0.0584	27.32	0.8451
	DIP	0.0748	25.18	0.7907	0.0626	26.72	0.8415	0.0512	28.47	0.8834
	Proposed	**0.0418**	**30.22**	**0.9153**	**0.0323**	**32.46**	**0.9508**	**0.0285**	**33.56**	**0.9614**

TABLE X
RECOVERY RESULTS OF LIGHT FIELD IMAGE *LEGO TRUCK* UNDER DIFFERENT ORS. RRSES AND PSNRS ARE LISTED. THE BEST VALUES ARE IN BOLD FACE. THE RESULTS OF LIGHT FIELD IMAGES *LEGO KNIGHTS*, *TAROT CARDS AND CRYSTAL BALL*, AND *THE STANFORD BUNNY* ARE LISTED IN THE SUPPLEMENTARY MATERIAL

Light field image	OR	0.05		0.1		0.15	
	Method	RRSE	PSNR (dB)	RRSE	PSNR (dB)	RRSE	PSNR (dB)
Lego Truck (200×200×3×8×8)	FBCP	0.1916	29.30	0.1700	30.34	0.1526	31.27
	FBCP-MP	0.2642	26.50	0.2352	27.51	0.2173	28.20
	HaLRTC	0.5250	20.54	0.4309	22.26	0.3486	24.10
	HTNN-FFT	0.2290	27.75	0.1553	31.12	0.1153	33.71
	HTNN-DCT	0.2336	27.57	0.1633	30.68	0.1241	33.06
	Proposed	**0.0843**	**36.43**	**0.0601**	**39.39**	**0.0405**	**42.79**

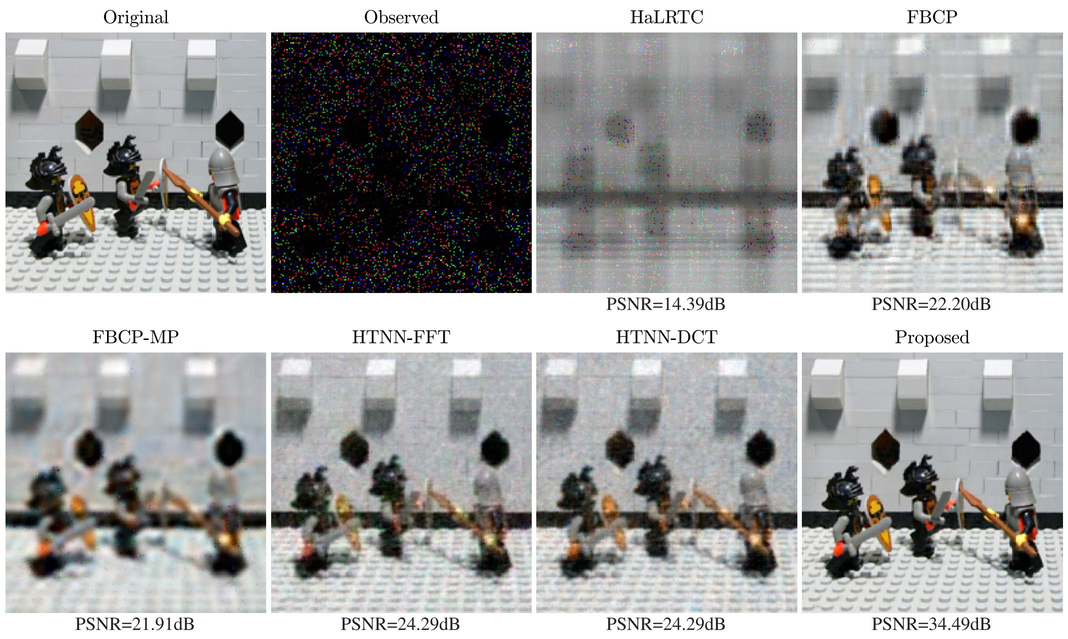

Fig. 11. Inpainting results of *Lego Knights* at OR = 0.05.

the CNet achieves excellent results. Besides, the fifth frames of the recovered *News* produced by all algorithms at OR = 0.05 are shown in Fig. 10(c). Again, the CNet is the best. Furthermore, CNet performs much better on *Bridge_close* and *News* than on *Bus*. This may be because the scenes of *Bridge_close* and *News* are quasi-static, which is consistent with low-rank latent modeling, whereas the *Bus*'s scenes are relatively dynamic in nature.

D. Light Field Images

We further investigate the performance of the CNet on light field images *Lego Knights*, *Lego Truck*, *Tarot Cards and Crystal Ball*, and *The Stanford Bunny* from The (New) Stanford Light Field Archive[6], each of which contains 289 color images captured from different views on a 17×17 grid. To reduce the computational cost, we resize all these color images to dimensions 200×200 and select a subset of images on a continuous 8×8 grid. Thus, each light field image is represented by a fifth-order tensor of dimensions 200×200×3×8×8. Similar to the color video situation, for fifth-order tensors, the encoder consists of the five-branch MLPs. We compare our method

[6]http://lightfield.stanford.edu/lfs.html

with FBCP, FBCP-MP, HaLRTC, HTNN-FFT, and HTNN-DCT, which are able to handle fifth-order tensors. The completion results are listed in Table X, which shows the excellent performance of CNet. We have tried to run DIP on a GPU with 32G memory. However, the memory is still not enough to perform DIP. Therefore, we have not compared CNet with DIP, which is inefficient for fifth-order TC.

VI. Conclusion

In this paper, we devised an encoder-decoder network for estimating missing values of visual data and demonstrated its excellent performance via experiments using color images, grayscale videos, HSIs, color videos, and light field images. From the recovered results, we see the powerful inpainting abilities of the proposed neural network. The purpose of introducing the decoder is to improve the quality of visual data. To achieve this goal, we designed CNN comprising two downsamplings, two upsamplings, and two channel compression operations, along with skip connections. However, there are a number of different options available for the structure of the decoder, like Transformer, which may yield even better performance. Besides, beyond CP decomposition, tensor completion neural networks based on other tensor decompositions may be promising.

References

[1] M. Mørup, "Applications of tensor (multiway array) factorizations and decompositions in data mining," *Interdiscipl. Rev., Data Mining Knowl. Discovery*, vol. 1, no. 1, pp. 24–40, Jan. 2011.

[2] Y. Song, Z. Gong, Y. Chen, and C. Li, "Tensor-based sparse Bayesian learning with intra-dimension correlation," *IEEE Trans. Signal Process.*, vol. 71, pp. 31–46, 2023.

[3] Q. Li and D. Schonfeld, "Multilinear discriminant analysis for higher-order tensor data classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 12, pp. 2524–2537, Dec. 2014.

[4] T.-X. Jiang, X.-L. Zhao, H. Zhang, and M. K. Ng, "Dictionary learning with low-rank coding coefficients for tensor completion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 2, pp. 932–946, Feb. 2023.

[5] A. Shashua and T. Hazan, "Non-negative tensor factorization with applications to statistics and computer vision," in *Proc. Int. Conf. Mach. Learn.,* Bonn, Germany, Aug. 2005, pp. 792–799.

[6] Y. Chen, Z. Liu, and Z. Zhang, "Tensor-based human body modeling," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Portland, OR, USA, Jun. 2013, pp. 105–112.

[7] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 208–220, Jan. 2013.

[8] J. A. Bengua, H. N. Phien, H. D. Tuan, and M. N. Do, "Efficient tensor completion for color image and video recovery: Low-rank tensor train," *IEEE Trans. Image Process.*, vol. 26, no. 5, pp. 2466–2479, May 2017.

[9] J. Miao and K. I. Kou, "Quaternion-based bilinear factor matrix norm minimization for color image inpainting," *IEEE Trans. Signal Process.*, vol. 68, pp. 5617–5631, 2020.

[10] L. Chen, X. Jiang, X. Liu, and M. Haardt, "Reweighted low-rank factorization with deep prior for image restoration," *IEEE Trans. Signal Process.*, vol. 70, pp. 3514–3529, 2022.

[11] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3551–3582, Jul. 2017.

[12] J. D. Carroll and J. Chang, "Analysis of individual differences in multidimensional scaling via an N-way generalization of 'Eckart-Young' decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, Sep. 1970.

[13] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, Aug. 2009.

[14] C. J. Hillar and L. H. Lim, "Most tensor problems are NP-hard," *J. ACM*, vol. 60, no. 6, pp. 1–39, Nov. 2013.

[15] S. E. Leurgans, R. T. Ross, and R. B. Abel, "A decomposition for three-way arrays," *SIAM J. Matrix Anal. Appl.*, vol. 14, no. 4, pp. 1064–1083, Oct. 1993.

[16] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, "Scalable tensor factorizations for incomplete data," *Chemom. Intell. Lab. Syst.*, vol. 106, no. 1, pp. 41–56, Mar. 2011.

[17] Q. Zhao, L. Zhang, and A. Cichocki, "Bayesian CP factorization of incomplete tensors with automatic rank determination," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1751–1763, Jan. 2015.

[18] T. Yokota, Q. Zhao, and A. Cichocki, "Smooth PARAFAC decomposition for tensor completion," *IEEE Trans. Signal Process.*, vol. 64, no. 20, pp. 5423–5436, Oct. 2016.

[19] Y. Xu, R. Hao, W. Yin, and Z. Su, "Parallel matrix factorization for low-rank tensor completion," *Inverse Probl. Imag.*, vol. 9, no. 2, pp. 601–624, May 2015.

[20] A. Cichocki, "Tensor networks for big data analytics and large-scale optimization problems," Jul. 2014, *arXiv:1407.3124.*

[21] I. V. Oseledets, "Tensor-train decomposition," *SIAM J. Sci. Comput.*, vol. 33, no. 5, pp. 2295–2317, 2011.

[22] Q. Zhao, G. Zhou, S. Xie, L. Zhang, and A. Cichocki, "Tensor ring decomposition," Jun. 2016, *arXiv:1606.05535*.

[23] W. Wang, V. Aggarwal, and S. Aeron, "Efficient low rank tensor ring completion," in *Proc. IEEE Int. Conf. Comput. Vis.*, Venice, Italy, Oct. 2017, pp. 5697–5705.

[24] H. Huang, Y. Liu, J. Liu, and C. Zhu, "Provable tensor ring completion," *Signal Process.*, vol. 171, Jun. 2020, Art. no. 107486.

[25] J. Yu, G. Zhou, W. Sun, and S. Xie, "Robust to rank selection: Low-rank sparse tensor-ring completion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 5, pp. 2451–2465, May 2023.

[26] P. Zhou, C. Lu, Z. Lin, and C. Zhang, "Tensor factorization for low-rank tensor completion," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1152–1163, Mar. 2018.

[27] X. P. Li and H. C. So, "Robust low-rank tensor completion based on tensor ring rank via $\ell_{p,\epsilon}$-norm," *IEEE Trans. Signal Process.*, vol. 69, pp. 3685–3698, May 2021.

[28] Z. Zhang and S. Aeron, "Exact tensor completion using t-SVD," *IEEE Trans. Signal Process.*, vol. 65, no. 6, pp. 1511–1526, Mar. 2017.

[29] M. E. Kilmer, K. Braman, N. Hao, and R. C. Hoover, "Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging," *SIAM J. Matrix Anal. Appl.*, vol. 34, no. 1, pp. 148–172, Feb. 2013.

[30] M. E. Kilmer and C. D. Martin, "Factorization strategies for third-order tensors," *Linear Algebra Its Appl.*, vol. 435, no. 3, pp. 641–658, Aug. 2011.

[31] L. Zhang, L. Song, B. Du, and Y. Zhang, "Nonlocal low-rank tensor completion for visual data," *IEEE Trans. Cybern.*, vol. 51, no. 2, pp. 673–685, Feb. 2021.

[32] C. Lu, X. Peng, and Y. Wei, "Low-rank tensor completion with a new tensor nuclear norm induced by invertible linear transforms," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Long Beach, CA, USA, Jun. 2019, pp. 5996–6004.

[33] E. Kernfeld, M. Kilmer, and S. Aeron, "Tensor–tensor products with invertible linear transforms," *Linear Algebra Appl.*, vol. 485, pp. 545–570, Aug. 2015.

[34] Y. Chen, X. Gui, J. Zeng, X.-L. Zhao, and W. He, "Combining low-rank and deep plug-and-play priors for snapshot compressive imaging," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 11, pp. 16396–16408, Nov. 2024.

[35] X.-L. Zhao, J.-H. Yang, T.-H. Ma, T.-X. Jiang, M. K. Ng, and T.-Z. Huang, "Tensor completion via complementary global, local, and nonlocal priors," *IEEE Trans. Image Process.*, vol. 31, pp. 984–999, 2021.

[36] X.-L. Zhao, W.-H. Xu, T.-X. Jiang, Y. Wang, and M. K. Ng, "Deep plug-and-play prior for low-rank tensor completion," *Neurocomputing*, vol. 400, pp. 137–149, Mar. 2020.

[37] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, "Plug-and-play image restoration with deep denoiser prior," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 6360–6376, Oct. 2022.

[38] H. Xiang, Q. Zou, M. A. Nawaz, X. Huang, F. Zhang, and H. Yu, "Deep learning for image inpainting: A survey," *Pattern Recognit.*, vol. 134, Feb. 2023, Art. no. 109046.

[39] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, Utah, USA, Jun. 2018, pp. 9446–9454.

[40] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 18–44, Feb. 2021.

[41] W. Qin, H. Wang, F. Zhang, J. Wang, X. Luo, and T. Huang, "Low-rank high-order tensor completion with applications in visual data," *IEEE Trans. Image Process.*, vol. 31, pp. 2433–2448, Mar. 2022.

[42] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan, "Tensor robust principal component analysis: Exact recovery of corrupted low-rank tensors via convex optimization," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, Jun. 2016, pp. 5249–5257.

[43] M. Ashraphijuo and X. Wang, "Fundamental conditions for low-CP-rank tensor completion," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 2116–2145, Jul. 2017.

[44] Z. Jin, M. Z. Iqbal, D. Bobkov, W. Zou, X. Li, and E. Steinbach, "A flexible deep CNN framework for image restoration," *IEEE Trans. Multimed.*, vol. 22, no. 4, pp. 1055–1068, Apr. 2020.

[45] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. 18th Int. Conf. Med. Image Comput. Comput.-Assist. Interv*, Munich, Germany, Oct. 2015, pp. 234–241.

[46] X. Hu, M. A. Naiel, A. Wong, M. Lamm, and P. Fieguth, "RUNet: A robust UNet architecture for image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Long Beach California, 2019, pp. 505–507.

[47] J. Gurrola-Ramos, O. Dalmau, and T. E. Alarcón, "A residual dense U-Net neural network for image denoising," *IEEE Access*, vol. 9, pp. 31742–31754, 2021.

[48] P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud, "Two deterministic half-quadratic regularization algorithms for computed imaging," in *Proc. IEEE Int. Conf. Image Process.*, vol. 2, Austin, TX, USA, Nov. 1994, pp. 168–172.

[49] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Fast and accurate image super-resolution with deep Laplacian pyramid networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2599–2613, Nov. 2019.

[50] Y. Luo, X.-L. Zhao, D. Meng, and T.-X. Jiang, "HLRTF: Hierarchical low-rank tensor factorization for inverse problems in multi-dimensional imaging," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Fukuoka, Japan, Oct. 2022, pp. 19303–19312.

[51] J.-L. Wang, T.-Z. Huang, X.-L. Zhao, Y.-S. Luo, and T.-X. Jiang, "CoNoT: Coupled nonlinear transform-based low-rank tensor representation for multidimensional image completion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 7, pp. 8969–8983, Jul. 2024.

[52] L. Chen, X. Jiang, X. Liu, and Z. Zhou, "Logarithmic norm regularized low-rank factorization for matrix and tensor completion," *IEEE Trans. Image Process.*, vol. 30, pp. 3434–3449, Mar. 2021.

[53] M. Yang, Q. Luo, W. Li, and M. Xiao, "3-D array image data completion by tensor decomposition and nonconvex regularization approach," *IEEE Trans. Signal Process.*, vol. 70, pp. 4291–4304, Aug. 2022.

[54] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[55] O. Sidorov and J. Y. Hardeberg, "Deep hyperspectral prior: Single-image denoising, inpainting, super-resolution," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop*, Seoul, Korea, Oct. 2019, pp. 3844–3851.

Xiang-Yu Wang received the B.E. degree from the Northwestern Polytechnical University, Xi'an, China, in 2021. He is currently working toward the Ph.D. degree with the Department of Electrical Engineering, City University of Hong Kong. His research interests include array signal processing and tensor analysis.

Xiao-Peng Li (Member, IEEE) received the B.Eng. degree as an outstanding graduate in electronic science and technology from Yanshan University, Qinhuangdao, China, in 2015, and the M.Sc. degree with Distinction in electronic information engineering and the Ph.D. degree in electrical engineering from the City University of Hong Kong, Hong Kong SAR, China, in 2018 and 2022, respectively. He was a Research Assistant with the Department of Information Engineering, Shenzhen University, Shenzhen, China from 2018 to 2019, and a Postdoctoral Fellow with the Department of Electrical Engineering, City University of Hong Kong from 2022 to 2023. Currently, he is an Assistant Professor with the College of Electronics and Information Engineering, Shenzhen University. His research interests include robust signal processing, sparse recovery, matrix processing, tensor processing, optimization methods, machine learning, and their applications in various areas of engineering, including target estimation, image recovery, video restoration, hyperspectral unmixing, and stock market analysis.

Nicholas D. Sidiropoulos (Fellow, IEEE) received the Diploma in electrical engineering from Aristotle University of Thessaloniki, Thessaloniki, Greece, and the M.S. and Ph.D. degrees in electrical engineering from the University of Maryland at College Park, College Park, MD, USA, in 1988, 1990, and 1992, respectively. He is the Louis T. Rader Professor with the Department of ECE, University of Virginia. He has previously served on the Faculty with the University of Minnesota and the Technical University of Crete, Greece. His research interests include in signal processing, communications, optimization, tensor decomposition, and machine learning. He received the NSF/CAREER award in 1998, the IEEE Signal Processing Society (SPS) Best Paper Award in 2001, 2007, 2011, and 2023, and the IEEE SPS Donald G. Fink Overview Paper Award in 2023. He served as an IEEE SPS Distinguished Lecturer (2008–2009), the Vice President-Membership (2017–2019), and the Chair of the SPS Fellow Evaluation Committee (2020–2021). He received the 2010 IEEE SPS Meritorious Service Award, the 2013 Distinguished Alumni Award of the ECE Department at the University of Maryland, the 2022 EURASIP Technical Achievement Award, and the 2022 IEEE SPS Claude Shannon-Harry Nyquist Technical Achievement Award. He is a fellow of EURASIP (2014).

Hing Cheung So (Fellow, IEEE) was born in Hong Kong. He received the B.Eng. degree from the City University of Hong Kong and the Ph.D. degree from The Chinese University of Hong Kong, in 1990 and 1995, respectively, both in electronic engineering. From 1990 to 1991, he was an Electronic Engineer with the Research and Development Division, Everex Systems Engineering Ltd., Hong Kong. From 1995 to 1996, he was a Postdoctoral Fellow with The Chinese University of Hong Kong. From 1996 to 1999, he was a Research Assistant Professor with the Department of Electronic Engineering, City University of Hong Kong, where he is currently a Professor. His research interests include detection and estimation, fast and adaptive algorithms, multidimensional harmonic retrieval, robust signal processing, source localization, and sparse approximation. He has been on the editorial boards of IEEE Signal Processing Magazine (2014–2017), IEEE TRANSACTIONS ON SIGNAL PROCESSING (2010–2014), *Signal Processing* (2010–), and *Digital Signal Processing* (2011–). He was also a Lead Guest Editor for IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING, special issue on "*Advances in Time/Frequency Modulated Array Signal Processing*" in 2017. In addition, he was an Elected Member in Signal Processing Theory and Methods Technical Committee (2011–2016) of the IEEE Signal Processing Society where he was the Chair in the Awards Subcommittee (2015–2016).