Enhanced Deep Unrolling Networks for Snapshot Compressive Hyperspectral Imaging

Xinran Qin[a], Yuhui Quan[a,b,*], Hui Ji[c]

[a]*School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China*
[b]*Pazhou Lab, Guangzhou 510335, China*
[c]*Department of Mathematics, National University of Singapore, 119076, Singapore*

Abstract

Snapshot compressive hyperspectral imaging necessitates the reconstruction of a complete hyperspectral image from its compressive snapshot measurement, presenting a challenging inverse problem. This paper proposes an enhanced deep unrolling neural network, called EDUNet, to tackle this problem. The EDUNet is constructed via the deep unrolling of a proximal gradient descent algorithm and introduces two innovative modules for gradient-driven update and proximal mapping reflectivity. The gradient-driven update module leverages a memory-assistant descent approach inspired by momentum-based acceleration techniques, for enhancing the unrolled reconstruction process and improving convergence. The proximal mapping is modeled by a subnetwork with a cross-stage spectral self-attention, which effectively exploits the inherent self-similarities present in hyperspectral images along the spectral axis. It also enhances feature flow throughout the network, contributing to reconstruction performance gain. Furthermore, we introduce a spectral geometry consistency loss, encouraging EDUNet to prioritize the geometric layouts of spectral curves, leading to a more precise capture of spectral information in hyperspectral images. Experiments are conducted using three benchmark datasets including KAIST, ICVL, and Harvard, along with some real data, comprising a total of 73 samples. The experimental results demon-

*Corresponding author at: School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China.

Email addresses: csqinxinran@gmail.com (Xinran Qin), csyhquan@scut.edu.cn (Yuhui Quan), matjh@nus.edu.sg (Hui Ji)

strate that EDUNet outperforms 15 competing models across four metrics including PSNR, SSIM, SAM, and ERGAS.

Keywords: Hyperspectral imaging, Snapshot compressive imaging, Image reconstruction, Deep unrolling networks

1. Introduction

Hyperspectral imaging aims at capturing a hyperspectral image (HSI) in the form of a 3D cube of intensities representing the integrals of radiance of a real scene across a wide range of spectral bands. HSIs provide rich spectral characteristics of objects or scenes, which are useful for finding objects, identifying materials, or detecting processes. As a result, hyperspectral imaging finds various applications in science and industry, such as remote sensing [1, 2], mineral exploration [3], medical diagnosis [4] and environment monitoring [5].

One key part in hyperspectral imaging is image reconstruction, *i.e.*, how to efficiently and accurately reconstruct HSIs from measurement data. This reconstruction process is tied with the scanning technology. There are many scanning technologies in hyperspectral imaging, and a popular one is the snapshot compressive spectral imaging, often referred to as coded aperture snapshot spectral imaging (CASSI) [6], which leverages compressive sensing for rapid and efficient acquisition of HSIs.

Different from traditional techniques that use sensor arrays to measure objects at multiple spectral bands, CASSI captures hyperspectral cubes in just a single coded 2D snapshot. This snapshot measures objects modulated by a physical mask and a disperser, creating a mixture of different wavelengths. Subsequently, a reconstruction algorithm is employed to reconstruct the 3D HSI from the 2D compressive snapshot.

The CASSI-based HSI reconstruction is a challenging ill-posed inverse problem. In recent years, deep learning has emerged as a prominent approach for developing powerful solutions to this problem; see *e.g.* [7, 8, 9, 10, 11, 12]. Many of these methods utilize neural networks (NNs) to model the inverse mapping from a snapshot to its corresponding HSI, which are then learned over extensive datasets. This paper focuses on tackling the CASSI-based HSI reconstruction problem.

1.1. Motivation and Main Idea

Among the existing designs of NN architectures, deep unrolling network (DUN) has become the most popular one for HSI reconstruction due to its capability of incorporating imaging physics as well as its interpretability, which can mitigate possible overfitting. A typical DUN unfolds an iterative scheme that solves a regularized variational model of hyperspectral imaging, where the regularization-related components are replaced by learnable NN modules. This unfolding process can be interpreted as a sequence of updating steps and refinement steps:

$$\cdot \xrightarrow{\text{Update}} \cdot \xrightarrow{\text{Refine}} \cdot \xrightarrow{\text{Update}} \cdot \xrightarrow{\text{Refine}} \cdots .$$

Despite extensive studies on DUNs, there is still a practical need for higher reconstruction accuracy. In this paper, we propose an enhanced DUN for CASSI-based HSI reconstruction, which exhibits improved performance compared to existing deep NNs.

Our proposed DUN is built upon the proximal gradient descent (PGD) algorithm [13, 14, 15], a widely used iterative numerical scheme for solving regularization models related to inverse problems in imaging. The PGD algorithm alternates between the following two steps:

1. A gradient descent step to update the image estimate;
2. A proximal mapping step for refining the estimate by exploiting specific characteristics of latent images.

Contrary to many existing DUNs that utilize half-quadratic splitting (HQS) (*e.g.* [8]) or the alternating direction method of multipliers (ADMM) (*e.g.* [16]), whose update step requires computing an exact solution of a large linear system arsing from the fidelity term of the unrolled model, our approach adopts a PGD-based update which only employs a single-step gradient descent. This simplification significantly reduces the computational complexity. Furthermore, our proposed DUN sets itself apart from existing models with three specific enhancements tailored for HSI reconstruction. These enhancements refine the main steps above and the training loss, offering substantial advantages over existing techniques. Detailed discussions of these enhancements are provided below.

Momentum inspired memory-assistant gradient-based update. In most existing DUNs, the updating step involves a predefined, non-learnable process, such as gradient-based updates. However, these updates are based on the first-order gradient which tends to follow a zig-zag direction, which slows down the movement towards the minima. Moreover, they exhibit slow convergence near the minima or saddle points, as the gradient magnitude rapidly vanishes in those regions. To address this, popular techniques like *momentum* are utilized, for instance, in RMSProp [17], Adam [18] and SUM [19]. Instead of relying solely on the current gradient, momentum accumulates the gradients from past steps to determine the direction of movement. This accumulation helps accelerate convergence by dampening zig-zag oscillations and building up speed towards the minima, allowing quicker convergence.

Motivated by the advantages of momentum for gradient-based updates, we introduce a novel approach that incorporates the concept of momentum into our DUN-based model for gradient-driven updates. To retain the effectiveness of momentum, which relies on the memory of past gradients, we design a neural block with a memory-assistant mechanism. This mechanism, implemented using convolutional long short-term memory (ConvLSTM) [20] allows utilizing gradient descents from previous stages. ConvLSTM not only leverages the gradients' memory but also potentially exploits the local structure of the measurement matrix, related to the update steps, providing additional benefits for our approach. By introducing the memory-assistant mechanism through ConvLSTM, our approach enhances the gradient-driven updates without compromising the interpretability of DUNs.

Cross-stage self-attention for refinement steps. An HSI exhibits specific physical characteristics, one of which is self-similarity and strong correlation along the spectral axis. The spectral axis entries represent the same object region at different wavelengths. To effectively exploit these characteristics for better performance and reduced overfitting, we propose a sub-NN equipped with a self-attention (SA) module along the spectral axis for the proximal mapping step. Although SA [21] is not a new concept, our design stands out from existing approaches by introducing a cross-stage mechanism.

The cross-stage SA module serves an additional purpose of exploiting the similarities among features learned at different stages. These similarities arise from the consistent role of the refinement step across various stages. By forming a path between two different stages, the cross-stage SA efficiently delivers features across stages and facilitates interactions among features from differ-

ent stages. The benefits of leveraging such feature similarities are twofold. First, it streamlines the feature delivery process across stages, promoting more efficient information flow and enhancing the model's ability to leverage valuable information throughout the DUN. Second, it brings certain regularization by establishing similarity relation of distinct stages.

Spectral geometry consistency loss for training HSI reconstruction NNs. In addition to the standard ℓ_1 loss, we propose a spectral geometry consistency loss for training the model. This loss encourages the model to focus more on the profile of spectral changes during the reconstruction process, effectively regularizing the NN and empirically improving reconstruction accuracy.

1.2. Contributions

See below for a summary of our technical achievements:

1. We introduce a momentum-motivated memory-assistant learnable module that effectively models the gradient descent steps of the PGD algorithm without compromising its interpretability.
2. We propose a cross-stage spectral self-attentive NN, specifically designed to model the proximal mapping. It enables the exploitation of characteristics of HSIs and enhances the efficiency of feature flow throughout the whole DUN.
3. Our training scheme incorporates a spectral geometry consistency loss, which effectively regularizes the learning process by leveraging the specific characteristics of HSIs.

These advancements lead to a lightweight DUN with superior performance over existing HSI reconstruction methods. As a result, this work not only enriches the technical aspects, but also broadens the practical applicability of CASSI-based hyperspectral imaging.

2. Related Work

2.1. Regularization-Based Methods with Pre-Defined Image Priors

Regularization, which imposes specific priors on HSIs, is a prevalent method in HSI reconstruction. Several representative image priors initially proposed for natural images in digital photography have been extended to HSIs. For instance, the sparsity prior [22] assumes that the gradients of

an HSI are sparse. The self-similarity prior [23, 24, 25, 26, 27] assumes that patches of an HSI tend to repeat themselves throughout the image. However, these priors often fall short of accurately representing HSIs with complex structures, limiting their effectiveness in high-accuracy reconstruction.

Recent studies have shifted from introducing image priors as variational models to embedding them within pre-trained denoising NNs. For instance, Plug-and-Play methods [22, 28, 29] utilize denoising NNs pre-trained on HSIs or natural images to regularize the reconstruction process. While these denoising NN-based priors show improved accuracy compared to pre-defined priors, their generalizability is constrained. This limitation stems from the fact that ideal priors used in reconstruction are typically aimed at mitigating residual errors during the reconstruction process, rather than random noise which is the primary focus of a denoising NN.

An alternative strategy involves self-supervised learning, as outlined in the studies [30, 31], which leverages the implicit image prior encoded in an untrained NN. This strategy employs an untrained NN to re-parameterize a latent HSI, subsequently training it to align with observed snapshot. Although this online learning approach is promising, it incurs significant computational costs due to the necessity of NN re-training for each snapshot in test time.

In comparison to all aforementioned methods, our approach leverages an end-to-end trained NN, which can learn powerful data-driven yet generalizable priors to handle unseen HSIs with complex structures, while avoiding the costly test-time training.

2.2. End-to-End Deep Learning-Based Methods

An increasingly prominent approach for HSI reconstruction is end-to-end training of a deep NN that maps a snapshot to the corresponding latent HSI. This approach has been widely explored in the literature; see *e.g.* [9, 32, 33, 34, 35, 36]. These studies primarily concentrate on the design of architecture, often neglecting the physical model of CASSI. This missing utilization of physics of CASSI can result in overfitting, particularly when the training data is not comprehensive enough. To utilize the physical model of CASSI for alleviating possible overfitting, some DUNs have been proposed to achieve physics awareness (*e.g.* [12, 37, 38]). A DUN typically consists of paired steps: one step for updating the estimate of the latent HSI and the other step for refining the estimate with a learnable prior. Most existing works focus on the

latter step, which can be viewed as a denoising NN exploiting various data-adaptive image priors, *e.g.*, spatial-spectral prior [8], non-local self-similarity prior [10], and patch-level Gaussian scale mixture prior [39]. In the following, we review the key techniques used in existing DUNs that are closely-related to our work.

Learning updating steps in DUNs. Zhang *et al.* [12] proposed a method that replaces the operators $\mathbf{\Phi}, \mathbf{\Phi}^\top$ involved in the gradient descent step of PGD, by convolutions and residual blocks. They incorporated channel attention to estimate the step size in PGD from the output of the previous stage. In contrast, motivated by the momentum-based acceleration, our approach differs in that we do not learn these operators but rather utilize them to achieve a more effective update step. It is worth noting that Mou *et al.* [40] used residual blocks to estimate gradient descent steps for natural image recovery. The methods above predominantly concentrate on independently optimizing the PGD step within each individual stage. They overlooked the potential advantages of integrating the information flow of gradient-based updating steps across various stages, which could significantly improve the PGD optimization process. In contrast, we employ ConvLSTM to exploit the inter-stage dependencies for an accurate estimation of the update step.

Self-attention for HSI reconstruction. Existing works on HSI reconstruction have extensively explored the use of SA. For instance, Miao *et al.* [9] utilized a generative adversarial network with SA for the initial stage. Meng *et al.* [41] employed three spatial-spectral SA modules to exploit the spatial-spectral correlation of HSIs, while Hu *et al.* [35] developed a spatial-spectral attention module with efficient feature fusion. Cai *et al.* [42] proposed a coarse-to-fine transformer, employing a spectra-aware screening mechanism for selecting coarse patches and using a customized spectra-aggregation hashing multi-head SA for fine pixel clustering and self-similarity capturing.

Unlike the aforementioned methods which define spatial features as tokens and ignore the correlation between feature channels, our approach adopts a different perspective that treats channel maps as tokens for SA, allowing effectively capturing inter-dependencies among different HSI channels which are critical for reconstruction. This idea aligns with a parallel work [36], which also treats spectral maps as tokens in a transformer-based model. However, this work as well as others primarily emphasize the correlations of features within the same stage while disregarding feature similarities across

various stages. In contrast, our approach differs in using SA in a cross-stage manner, significantly enhancing the feature flow throughout the DUN.

Loss functions for HSI reconstruction. Most existing NNs for HSI reconstruction are trained using the standard ℓ_2 or ℓ_1 loss. Hu *et al.* [35] introduced a frequency-domain loss to reduce the frequency-domain discrepancy between network predictions and ground truths. However, the spectral geometrical characteristics are usually ignored in these loss functions. In contrast, we introduce a loss function that focuses on narrowing the discrepancy in spectral geometric changes, bring improvement in performance and generalizability.

Memory DUNs for low-level vision. Song *et al.* [43, 44] proposed a memory-augmented DUN for compressed sensing, although it was not tested on HSI reconstruction. Their approach incorporated LSTM [45] into the denoising sub-networks during the refinement steps to enhance feature flow. Zhou *et al.* [46] presented a memory-augmented DUN for super-guided image super-resolution, where LSTM units and a non-local cross-modality module were employed to improve information representation. These works utilize memory modules for improving the feature flow of the refinement steps, but ignore the feature flow of the updating steps. In contrast, we utilize ConvLSTM for the update steps, not the refinement steps. Additionally, instead of using LSTM, we employ cross-stage SA to enhance feature flow in the refinement steps, which also effectively exploits the self-similarity of an HSI.

3. Proposed Approach

3.1. Problem Formulation

For ease of reference, Table 1 lists the symbols used for describing our approach. Let $\boldsymbol{X} \in \mathbb{R}^{M\times N\times\Lambda}$ denote an HSI with spatial indices m and n, and spectral index λ. Generally, the snapshot from a CASSI device can be expressed as follows [37, 41, 39]:

$$\boldsymbol{Y}(m,n) = \sum_{\lambda=1}^{\Lambda} \rho(\lambda)\varphi(m - J(\lambda), n)\boldsymbol{X}(m - J(\lambda), n, \lambda), \tag{1}$$

where $\boldsymbol{Y} \in \mathbb{R}^{M\times N}$ denotes the snapshot, $\rho(\cdot)$ the spectral response of the camera, $\varphi(\cdot,\cdot)$ the coded aperture pattern, and $J(\cdot)$ the dispersive function.

This expression can be re-formulated in a matrix-vector form while considering measurement noise:

$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x} + \boldsymbol{n}, \tag{2}$$

where $\boldsymbol{\Phi}$ denotes the measurement matrix determined by ρ and ψ, $\boldsymbol{n}$ the measurement noise, and $\boldsymbol{x}, \boldsymbol{y}$ the vectorized form of $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{N}$, respectively. As the linear system (2) is under-determined, HSI reconstruction requires solving an ill-posed linear inverse problem.

Table 1: Symbol list for proposed approach.

$\boldsymbol{X}$	Latent HSI	$\boldsymbol{h}^{(k)}$	Hidden state of ConvLSTM
$\boldsymbol{Y}$	Snapshot	$\boldsymbol{c}^{(k)}$	Cell state of ConvLSTM
ρ	Spectral response	$\boldsymbol{f}_k$	Forget gate vector of ConvLSTM
φ	Coded aperture pattern	$\boldsymbol{i}^{(k)}$	Input gate vector of ConvLSTM
J	Dispersive function	$\boldsymbol{o}^{(k)}$	Output gate vector of ConvLSTM
$\boldsymbol{\Phi}$	Measurement matrix	$\boldsymbol{g}^{(k)}$	Intermediate result in cell state
$\boldsymbol{x}$,$\boldsymbol{y}$	Vectorized form of $\boldsymbol{X}$,$\boldsymbol{Y}$	$\boldsymbol{W}$	Convolutional layer kernel
$\boldsymbol{n}$	Measurement noise	$\boldsymbol{b}$	Bias term
$\mathcal{R}$	Regularization function	$\boldsymbol{K}$	Keys in SA
λ	Regularization weight	$\boldsymbol{Q}$	Queries in SA
$\gamma^{(k)}$	Step size in PGD	$\boldsymbol{V}$	Values in SA
$\boldsymbol{u}^{(k)}$	Intermediate variable after GD	d	Length of a key/query/value vector
$\boldsymbol{x}^{(k)}$	Intermediate estimate of HSI	$\mathcal{D}$	Geometry map
Prox	Proximal mapping	γ	Weight of SGC loss
$\boldsymbol{x}'$	Generic vectorized HSI	α	Threshold value in spectral dimension
$\mathbb{R}^+$	Set of positive real numbers	$\boldsymbol{M}_{\boldsymbol{X}}$	Mask constructed for $\boldsymbol{X}$
$\boldsymbol{u}^{(k)}$	Gradient map	$\widehat{\boldsymbol{X}}$	Ground-truth HSI

3.2. Network Architecture via Deep Unrolling

Our proposed NN, named EDUNet (Enhanced Deep Unrolling Network), is constructed via unrolling the PGD solver of the following optimization model:

$$\min_{\boldsymbol{x}} \quad \|\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}\|_2^2 + \lambda\mathcal{R}(\boldsymbol{x}), \quad \lambda \in \mathbb{R}^+, \tag{3}$$

where $\mathcal{R}$ is a functional for regularization. The PGD algorithm [13, 14, 15] consists of two alternative steps: the gradient descent (GD) step, used for updating the estimate; and the proximal mapping (PM) step, employed to refine the estimate by fitting the functional $\mathcal{R}$ with an image prior. The PGD

iterations are performed as follows: for $k = 1, \cdots, K$,

$$\text{GD: } \boldsymbol{u}^{(k)} = \boldsymbol{x}^{(k-1)} + \gamma^{(k)} \boldsymbol{\Phi}^\top (\boldsymbol{y} - \boldsymbol{\Phi} \boldsymbol{x}^{(k-1)}), \tag{4}$$

$$\text{PM: } \boldsymbol{x}^{(k)} = \underset{\boldsymbol{x}'}{\operatorname{argmin}} \, \|\boldsymbol{x} - \boldsymbol{u}^{(k)}\|_2^2 + 2\gamma^{(k)} \mathcal{R}(\boldsymbol{u}^{(k)}), \tag{5}$$

where $\gamma^{(k)}$ denotes the step size and λ denotes the hyper-parameter for balancing regularization and fitting. These two steps are then mimicked by NN modules, forming the EDUNet.

Remark 1. *Various numerical methods are available for solving Eq.* (3)*, including HQS [8] and ADMM [16]. In these methods, the update step necessitates finding an exact solution to the linear system associated with the matrix* $\boldsymbol{\Phi}^\top \boldsymbol{\Phi}$*, a process that is time-consuming due to the large size of* $\boldsymbol{\Phi}^\top \boldsymbol{\Phi}$*. Even an approximate solution via iterative solvers demands extensive iterations and significant time. Conversely, the PGD-based update, leveraging single-step gradient descent, provides a markedly more efficient computational alternative. Further details on this approach can be found in existing literatures (e.g. [47]).*

Concretely, the GD step (4) is mimicked by the Memory-Assistant Descent (MAD) block, and the PM step (5) is mimicked by the Cross-stage Attentive Proximal (CAP) sub-network. As a result, the EDUNet is composed of K stages, each containing a MAD block for gradient-driven updating and a CAP sub-network for refinement/denoising. See Figure 1 for an overview of EDUNet. The EDUNet accepts the snapshot measurement $\boldsymbol{y}$ and the measurement matrix $\boldsymbol{\Phi}$ as input, passes them to the K stages, and uses the reconstructed HSI from the last stage as its final output: $\hat{\boldsymbol{x}} = \boldsymbol{x}^{(K)}$.

Inspired by the momentum-based acceleration technique widely used in gradient descent-based methods, the MAD block leveraging ConvLSTM acts as a gradient descent mechanism across different stages. This results in a more efficient update process compared to solely relying on the first-order gradient at the current stage. At the same time, the CAP sub-network leverages self-similarities present in an HSI via a cross-stage SA module, enabling the exploitation of special characteristics unique to HSIs and facilitating the rapid flow of features through the whole DUN.

Remark 2. *Most existing DUNs mainly focus on modeling the PM step* (5) *using a deep sub-NN to learn a data-driven prior. The GD step* (4) *is generally kept unchanged, with* $\gamma^{(k)}$ *being either a fixed or learnable parameter. Such a design is not optimal for reconstruction.*

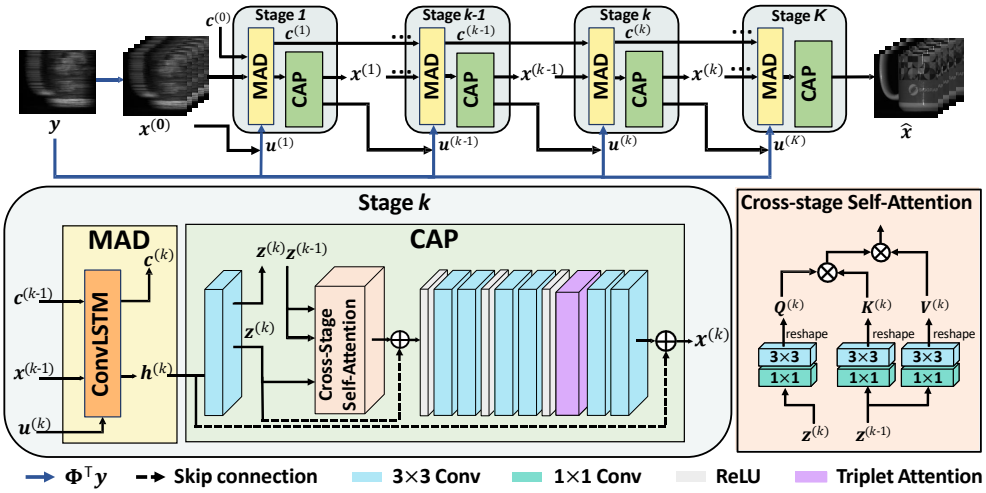

Figure 1: Overview of proposed EDUNet for CASSI-based HSI reconstruction.The overall architecture consists of K stages (top), each consisting of a MAD block that mimics the GD step (4) and a CAP sub-network that mimics the PM step (5).

Remark 3. *Many existing DUNs are developed by incorporating sub-NNs in an optimization algorithm to neutralize the refinement steps associated with the regularization term. In contrast, our approach also focuses on neutralizing the updating step associated with the fidelity term by employing a learnable NN module, leading to more-effective updates. This NN module can be seen as a neuralization of the momentum-based acceleration technique commonly used in gradient descent methods. Note that our approach is not the first to replace the updating step with an NN; previous works like [9] and [41] have also explored this idea. However, the key distinction lies in our approach being the first to utilize momentum-based acceleration, as well as specific characteristics of the problem, in the updating step.*

3.3. Memory-Assistant Descent Blocks

The MAD blocks consist of a series of ConvLSTM units [20] positioned at each stage of the NN. These ConvLSTM units utilize long-range dependencies among all cascading stages to assist in momentum-driven gradient updates. Within each MAD block at the k-stage, the gradient map $\boldsymbol{u}^{(k)}$ is defined by:

$$\boldsymbol{u}^{(k)} = \boldsymbol{\Phi}^\top(\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}^{(k-1)}) \tag{6}$$

where $\boldsymbol{x}^{(k-1)}$ denotes the estimate from the previous stage. The gradient map $\boldsymbol{u}^{(k)}$ is taken as input for the k-th ConvLSTM unit, introducing information

on gradient descent. Each ConvLSTM unit comprises a hidden state, denoted as $\boldsymbol{h}^{(k)}$, and a cell state, denoted as $\boldsymbol{c}^{(k)}$, at the k-th stage. The hidden state $\boldsymbol{h}^{(k)}$ has the same size as $\boldsymbol{x}^{(k)}$. The MAD block is defined as follows:

$$[\boldsymbol{h}^{(k)}, \boldsymbol{c}^{(k)}] = \text{ConvLSTM}(\boldsymbol{u}^{(k)}, \boldsymbol{x}^{(k-1)}, \boldsymbol{c}^{(k-1)}), \tag{7}$$

for $k = 1, \cdots, K$,

In contrast to the original ConvLSTM units, which use the previous hidden state $\boldsymbol{h}^{(k-1)}$ as input, we replace $\boldsymbol{h}^{(k-1)}$ with $\boldsymbol{x}^{(k-1)}$, which is the output from the CAP sub-network of the previous stage. The motivation behind this change is to utilize the current gradient descent defined over $\boldsymbol{x}^{(k-1)}$. The resulting $\boldsymbol{h}^{(k)}$ is then used as input for the CAP sub-network, while $\boldsymbol{c}^{(k)}$ is fed into the MAD block at the next stage, serving as an accumulator of state information.

In the k-th stage, the ConvLSTM unit calculates $\boldsymbol{h}_k$, $\boldsymbol{c}_k$ by the LSTM rules [45] as follows:

$$\boldsymbol{c}^{(k)} = \boldsymbol{f}_k \odot \boldsymbol{c}^{(k-1)} + \boldsymbol{i}^{(k)} \odot \tanh(\boldsymbol{g}^{(k)}), \tag{8}$$

$$\boldsymbol{h}^{(k)} = \boldsymbol{o}^{(k)} \odot \tanh(\boldsymbol{c}^{(k)}), \tag{9}$$

where $\odot$ denotes Hadamard product, and $\boldsymbol{i}_k$, $\boldsymbol{f}_k$, $\boldsymbol{o}_k$, $\boldsymbol{g}_k$ denote the input gate, forget gate, output gate, and the intermediate result, respectively, which are calculated using the standard scheme of [20] as follows:

$$\boldsymbol{i}^{(k)} = \text{sigmoid}(\boldsymbol{W}_{\text{mi}} \otimes \boldsymbol{u}^{(k)} + \boldsymbol{W}_{\text{xi}} \otimes \boldsymbol{x}^{(k-1)} + \boldsymbol{b}_{\text{i}}), \tag{10}$$

$$\boldsymbol{f}^{(k)} = \text{sigmoid}(\boldsymbol{W}_{\text{mf}} \otimes \boldsymbol{u}^{(k)} + \boldsymbol{W}_{\text{xf}} \otimes \boldsymbol{x}^{(k-1)} + \boldsymbol{b}_{\text{f}}), \tag{11}$$

$$\boldsymbol{g}^{(k)} = \boldsymbol{W}_{\text{mg}} \otimes \boldsymbol{u}^{(k)} + \boldsymbol{W}_{\text{xg}} \otimes \boldsymbol{x}^{(k-1)} + \boldsymbol{b}_{\text{g}}, \tag{12}$$

$$\boldsymbol{o}^{(k)} = \text{sigmoid}(\boldsymbol{W}_{\text{mo}} \otimes \boldsymbol{u}^{(k)} + \boldsymbol{W}_{\text{xo}} \otimes \boldsymbol{x}^{(k-1)} + \boldsymbol{b}_{\text{o}}), \tag{13}$$

where $\otimes$ denotes a convolutional layer with 3×3 kernels $\boldsymbol{W}_{**}$, and $\boldsymbol{b}_*$ denotes a bias term.

3.4. Cross-stage Self-Attentive Proximal Sub-Networks

The CAP block serves as a learnable PM step (5), refining the estimate obtained from the MAD block. It can also be perceived as a denoising NN, with the estimation residual being interpreted as noise.

Starting with $\boldsymbol{h}^{(k)}$ (of the same size as $\boldsymbol{x}$) from the MAD block as input, the CAP block maps it to a feature tensor $\boldsymbol{z}^{(k)}$ using a convolutional layer.

This tensor is then processed by a cross-stage SA module. Subsequently, the results are passed through a sequence of convolutional layers with rectified linear units (ReLUs), and a triplet attention mechanism [48] is integrated to better exploit spatial-channel dependencies. The output, having the same size as $\boldsymbol{x}$, is combined with the input $\boldsymbol{h}^{(k)}$ through a skip connection, ultimately generating the reconstructed HSI $\boldsymbol{x}^{(k)}$ at the current stage. See Figure 1 for the details.

Recall that SA [21] establishes relationship between input feature tokens to compute a refined feature representation. It begins by generating a key/query/value vector of length d from each token, and these vectors are collectively stored as $\boldsymbol{K}$, $\boldsymbol{Q}$, and $\boldsymbol{V}$, respectively. The SA operation is then calculated as follows:

$$\mathrm{SA}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \mathrm{softmax}\big(\frac{1}{\sqrt{d}}\boldsymbol{Q}\boldsymbol{K}^{\top}\big)\boldsymbol{V}, \tag{14}$$

where $\sqrt{d}$ is used as a normalization factor for the stability of training. We treat each feature channel as a token so as to exploit the self-similarities among feature channels. These tokens are aligned due to natural alignment of spectral slices of an HSI. In the kth stage, rather than use the feature $\boldsymbol{z}^{(k)}$ at current stage to calculate $\boldsymbol{K}^{(k)}, \boldsymbol{Q}^{(k)}, \boldsymbol{V}^{(k)}$, we only use $\boldsymbol{z}^{(k)}$ for $\boldsymbol{Q}^{(k)}$ while using the feature $\boldsymbol{z}^{(k-1)}$ of previous stage for $\boldsymbol{K}^{(k)}, \boldsymbol{V}^{(k)}$. Concretely, we calculate

$$\boldsymbol{Q}^{(k)} = \boldsymbol{W}_{\mathrm{Qd}}^{(k)} \otimes \boldsymbol{W}_{\mathrm{Qp}}^{(k)} \otimes \boldsymbol{z}^{(k)}, \tag{15}$$

$$\boldsymbol{K}^{(k)} = \boldsymbol{W}_{\mathrm{Kd}}^{(k)} \otimes \boldsymbol{W}_{\mathrm{Kp}}^{(k)} \otimes \boldsymbol{z}^{(k-1)}, \tag{16}$$

$$\boldsymbol{V}^{(k)} = \boldsymbol{W}_{\mathrm{Vd}}^{(k)} \otimes \boldsymbol{W}_{\mathrm{Vp}}^{(k)} \otimes \boldsymbol{z}^{(k-1)}, \tag{17}$$

where $\boldsymbol{W}_{(*p)}^{(k)}, \boldsymbol{W}_{(*d)}^{(k)}$ correspond to 1×1 convolutions and 3×3 depth-wise convolutions respectively for encoding spatial-channel context.

The motivation behind the cross-stage strategy is as follows. In the DUN architecture, there is an alternating scheme between the update and refinement stages. As the CAP sub-networks at different stages fulfill the role of refinement, the features they extract should exhibit a high correlation. Additionally, the features extracted from the previous stage can serve as good initializations for the corresponding features at the next stage. However, the aforementioned existing pipeline does not leverage such correlations for more efficient training, which can potentially create a bottleneck for features as they flow through the entire DUN.

To address this limitation, our proposed cross-stage SA scheme establishes a direct path between two stages, improving the flow of features across stages. This enables efficient feature transmission and enhances feature interactions during inference. Moreover, it establishes a relation between two adjacent stages, providing implicit regularization during training. This leads to more effective and smoother feature propagation within the DUN, facilitating better overall performance.

The multi-head strategy [21] is adopted for the cross-stage SA. We split the key/query/value matrices into H heads: $\boldsymbol{Q}^{(k)} = [\boldsymbol{Q}_1^{(k)}, \cdots, \boldsymbol{Q}_H^{(k)}]$, $\boldsymbol{K}^{(k)} = [\boldsymbol{K}_1^{(k)}, \cdots, \boldsymbol{K}_H^{(k)}]$, and $\boldsymbol{V}^{(k)} = [\boldsymbol{V}_1^{(k)}, \cdots, \boldsymbol{V}_H^{(k)}]$, along channel dimension. Then, the output is calculated as

$$\boldsymbol{O}^{(k)} = \bigcup_{h=1}^{H} \mathrm{SA}(\boldsymbol{Q}_h^{(k)}, \boldsymbol{K}_h^{(k)} \boldsymbol{V}_h^{(k)}), \tag{18}$$

where $\bigcup$ denotes concatenation. Afterward, $\boldsymbol{O}^{(k)}$ is reshaped for subsequent processing.

3.5. Loss Functions for Training

To enhance the generalization of a deep NN for HSI reconstruction, we propose the use of an auxiliary loss termed the Spectral Geometry Consistency (SGC) loss, which is motivated by the physical characteristics of HSI images. For an HSI $\boldsymbol{X} \in \mathbb{R}^{M \times N \times \Lambda}$, we define the geometry map $\mathcal{D}(\boldsymbol{X})$ by

$$\mathcal{D}(\boldsymbol{X}) = \nabla_{\mathrm{c}}(\mathrm{sign}(\nabla_{\mathrm{c}} \boldsymbol{X})) \in \{-1, 0, 1\}^{M \times N \times \Lambda}, \tag{19}$$

where ∇_c calculates the gradient along the spectral axis, and $\mathrm{sign}(\cdot)$ denotes the element-wise sign function. For a spatial location (m_0, n_0), $\mathcal{D}(\boldsymbol{X})[m_0, n_0, \cdot]$ indicates the wavelengths where the monotony of spectral values changes, representing an inherent geometrical property of the spectral curve. Leveraging $\mathcal{D}$, the SGC loss places emphasis on ensuring the geometrical layout consistency between the reconstructed HSI and the ground truth. This additional loss aids in promoting more accurate spectral structure preservation during the training process, leading to improved generalization capabilities of the NN for HSI reconstruction.

Considering HSIs exhibit high spatial sparsity, the irrelevant dark regions are omitted for robustness. This is achieved by constructing a mask $\boldsymbol{M}_{\boldsymbol{X}}$ that thresholds the maximal density along the spectral dimension:

$\boldsymbol{M}_{\boldsymbol{X}}(m,n,\lambda)=1$ if $\max_\lambda \boldsymbol{X}(m,n,\lambda)\geq\alpha$; and 0 otherwise. Then, the SGC loss is defined as

$$\mathcal{L}_{\text{sgc}} \triangleq \|\boldsymbol{M}_{\boldsymbol{X}}\odot\mathcal{D}(\boldsymbol{X})-\boldsymbol{M}_{\widehat{\boldsymbol{X}}}\odot\mathcal{D}(\widehat{\boldsymbol{X}})\|_1. \tag{20}$$

where $\boldsymbol{X},\widehat{\boldsymbol{X}}$ denote the reconstructed HSI and its ground truth respectively, and $\mathcal{D}(\boldsymbol{X}),\mathcal{D}(\widehat{\boldsymbol{X}})$ denote their respective geometry maps.

By minimizing $\mathcal{L}_{\text{sgc}}$, the predicted HSI is encouraged to align with the ground truths in terms of wavelength-density trends, effectively mitigating potential overfitting. The overall loss, incorporating a weighted factor $\gamma\in\mathbb{R}^+$, is then defined as:

$$\mathcal{L}=\|\boldsymbol{X}-\widehat{\boldsymbol{X}}\|_1+\gamma\|\boldsymbol{M}_{\boldsymbol{X}}\odot\mathcal{D}(\boldsymbol{X})-\boldsymbol{M}_{\widehat{\boldsymbol{X}}}\odot\mathcal{D}(\widehat{\boldsymbol{X}})\|_1. \tag{21}$$

where $\boldsymbol{X},\widehat{\boldsymbol{X}}$ denote the reconstructed HSI and its ground truth respectively. Additionally, $\mathcal{D}(\boldsymbol{X})$ and $\mathcal{D}(\widehat{\boldsymbol{X}})$ denote their respective geometry maps, and $\boldsymbol{M}_{\boldsymbol{X}}$ and $\boldsymbol{M}_{\widehat{\boldsymbol{X}}}$ denote their respective masks. The parameter γ is a hyper-parameter for balancing the fidelity term and the SGC loss.

4. Experiments

We implement EDUNet with PyTorch [49]. Throughout all the experiments, we consistently use the same set of hyper-parameters to ensure uniformity and comparability in the results. See Table 2 for a summary of the hyper-parameter setting of EDUNet. Unless specified, the kernel sizes are all set to 3×3 on all convolutional layers, and all the strides and padding sizes are set to 1. The stage number K is set to 6. The head number H for the SA in CAP blocks is set to 8. Regarding the training loss, we set $\alpha=\frac{5}{255}$ for $\boldsymbol{M}_{\boldsymbol{X}}$ as pixels with intensities less than 5 are usually perceived as dark pixels, and we set $\gamma=0.1$ for Eq. (21) to make the two terms in $\mathcal{L}$ have the same scale. We initialize $\boldsymbol{x}^{(0)}$ by $\boldsymbol{\Phi}^\top\boldsymbol{y}$ and the model weights by the Kaiming method. The training is done via the Adam optimizer with a fixed learning rate of 10^{-4}, batch size of 4, and a maximal epoch number of 200. The same data augmentation scheme as [39] is adopted, including rotation and flipping. The training process converges after 180 epochs. Our code will be released on GitHub upon the paper's acceptance.

Following [39], Peak-Signal-to-Ratio (PSNR) and Structured SIMilarity (SSIM) index are adopted as the metrics for quantitative evaluation. In addition, following [50, 10, 51], spectral angle mapper (SAM) and relative

Table 2: Hyper-parameters used in the proposed method.

NN		Training	
Stage Number K	6	α	5/255
Kernel Size	3×3	γ	0.1
Stride	1	Learning Rate	10^{-4}
Padding Size	1	Epcoh Number	200
Head Number H	8	Batch Size	4

dimensionless global error in synthesis (ERGAS) are also used as the quantitative metrics. The performance evaluation is conducted using three settings of training/test data. Table 3 presents a summary of the characteristics of the training/test data. See also Figure 2 for some samples from different datasets. In all experiments, the results of the compared baseline methods are quoted from existing literature if applicable. When retraining is necessary, we meticulously adhere to the hyperparameters specified in the original publications, for maintaining the integrity of our comparative analysis. For the other hyperparameters not specified in the original publications, we make the effort to report the best performance we can have.

Table 3: Training and test datasets of all experiments.

Training Dataset				Test Dataset			
Name	Size	#	Wavelength	Name	Size	#	Wavelength
CAVE	$256 \times 256 \times 28$	32	450-650nm	KAIST	$256 \times 256 \times 28$	10	450-650nm
ICVL	$1300 \times 1392 \times 31$	151	400-700nm	ICVL	$256 \times 256 \times 31$	50	400-700nm
Harvard	$512 \times 512 \times 31$	41	400-700nm	ICVL	$256 \times 256 \times 31$	9	400-700nm

4.1. Evaluation on Synthetic Data

4.1.1. CAVE and KAIST datasets

Following [41, 39], we use the 32 HSIs of the CAVE dataset [52] for training, and the 10 HSIs of the KAIST dataset [22] for test. Same as [41, 39] for a fair comparison, all these HSIs are cropped into patches with a spatial size of 256×256 and reduced to 28 wavelengths ranging from 450nm to 650nm via spectral interpolation. The snapshot measurements are generated by the 256×256 mask of CASSI used in [41].

Fifteen existing methods are chosen for comparison, including (a) one conventional method: DeSCI [24]; (b) one self-supervised deep learning-based

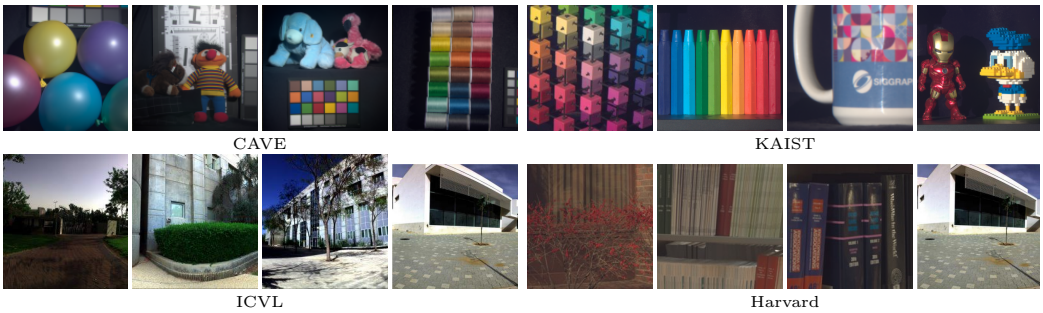

Figure 2: Samples from four datasets. As HSIs are difficult to visualize due to their cube form, we show the RGB references of the samples.

method: PnP-DIP [30]; and (c) twelve supervised learning-based methods: λ-Net [9] HSSP [8], DNU [10], TSA-Net [41], ADMMNet [37], GAP-Net [44], DGSMP [39], MADUN [43], MAPUN [44], HDNet [35], HDNet-SINR [53], MST-L [36] and CST-L [42]. The HSSP, DNU, GAP-Net, DGSMP and MADNU and MAPUN are based on DUNs. The HDNet, MST-L and CST-L are from the three latest works accepted in a very recent conference.

The quantitative results are listed in Table 4 , which are quoted from [36, 35] whenever possible. It can be seen that our approach significantly outperforms the compared ones. Specifically, EDUNet shows remarkable superior performance over other DUNs. It also surpasses CST-L, MST-L and HDNet (*i.e.* three latest methods) with an average PSNR gain of more than 0.2dB, 1dB, and 2dB, respectively. Also note that compared to MADUN and MAPUN which use LSTM for the denoising NN, EDUNet has a noticeable PSNR gain. See Table 5 for the results in terms of SAM and ERGAS, where the methods with released pre-trained models are used for comparison. Our EDUNet still performs the best among the compared methods.

Table 6 compares the model complexity of different methods according to the number of parameters, number of Giga floating-point operations (GFLOPs) and overall running time on a single NVIDIA GTX 1080 Ti GPU. Although EDUNet contains ConvLSTM and SA blocks, it is still kept compact to maintain a relatively-low model complexity. Among all compared methods, EDUNet has the smallest number of GFLOPs, and its size is smaller than all other models except DNU. Regarding running time, EDUNet is faster than DNU, DGSMP, MADNU, MAPUN, MST-L and CST-L, while comparable to TSA-Net. It is slower than λ-Net and HDNet, but with noticeably better performance. The reason is there is no advanced acceleration support

from GPU for the SA of EDUNet (same for TSA-Net, MST-L and CST-L), unlike standard convolutional layers. This is also one reason that EDUNet has the smallest number of GFLOPs but is not the fastest. To conclude, EDUNet achieves the best trade-off between performance and model complexity. Its performance gain is from architecture design rather than increase model complexity.

Following [41], we also evaluate performance on noisy cases. The measurements in training data are corrupted with 11-bit shot-noise while the ones in test data are corrupted with 11-bit,12-bit and 13-bit shot noise respectively. Table 7 lists the results of four recent methods and ours. All the compared models are retrained on the noisy measurements. It can be seen that our method still performs the best in noisy cases.

4.1.2. ICVL and Harvard Datasets

We also conduct experiments on the ICVL dataset [54] and the Harvard dataset [55], respectively. The ICVL dataset consists of 201 HSIs of real-world objects, each with a spatial size of 1300×1392, 31 spectral bands collected from 400nm to 700 nm at a 10nm step. The Harvard dataset consists of 50 outdoor scenes, each with a spatial size of 512×512, 31 spectral bands collected from 420nm to 720nm at a 10nm step. Following the protocol of [8, 56], 50 HSIs in the ICVL dataset and 9 HSIs in the Harvard dataset are used for test respectively, and the rest samples are used for training. Same as [8, 56] for fair comparison, all HSIs for training and test are cropped into patches with a spatial size of 48×48, while keeping the band number unchanged. The snapshot measurements are generated by the 48×48 mask of CASSI used in [8].

Six existing methods are selected for comparison, including (a) two conventional methods: SSNR [23] and ADLTR [27]; (b) six supervised learning-based methods: HSCNN [7], λ-Net [9], DNU [10], DTLP [56], HDNet [35] and CST-L [42]. DNU and DTLP use DUNs, HDNet and CST-L are the latest two methods. Following [56], two additional quantitative metrics including SAM and ERGAS are introduced for evaluation.

See Table 8 for quantitative comparison. The results of the compared methods are cited from [56], except for HDNet and CST-L. These two methods have no published results on the dataset and thus we train their models using their officially released codes, with their loss weights tuned for better performance. The proposed one outperformed all other methods, with more than 0.6db PSNR improvement on both datasets. Such noticeable per-

Table 4: Quantitative results in PSNR(dB) (even rows) and SSIM (odd rows) on KAIST dataset in a noiseless setting. Best results are boldfaced.

Method	Metrics	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	Mean
DeSCI	PSNR	27.13	23.04	26.62	34.96	23.94	22.38	24.45	22.03	24.56	23.59	25.27
	SSIM	0.748	0.62	0.818	0.897	0.706	0.683	0.743	0.673	0.732	0.587	0.721
λ-net	PSNR	30.1	28.49	27.73	37.01	26.19	28.64	26.47	26.09	27.5	27.13	28.53
	SSIM	0.849	0.805	0.87	0.934	0.817	0.853	0.806	0.831	0.826	0.816	0.841
HSSP	PSNR	31.48	31.09	28.96	34.56	28.53	30.83	28.71	30.09	30.43	28.78	30.35
	SSIM	0.858	0.842	0.823	0.902	0.808	0.877	0.824	0.881	0.868	0.842	0.852
DNU	PSNR	31.72	31.13	29.99	35.34	29.03	30.87	28.99	30.13	31.03	29.14	30.74
	SSIM	0.863	0.846	0.845	0.908	0.833	0.887	0.839	0.885	0.876	0.849	0.863
PnP-DIP	PSNR	32.68	27.26	31.3	40.54	29.79	30.39	28.18	29.44	34.51	28.51	31.26
	SSIM	0.890	0.833	0.914	0.962	0.900	0.877	0.913	0.874	0.927	0.851	0.894
TSA-Net	PSNR	32.03	31.00	32.25	39.19	29.39	31.44	30.32	29.35	30.01	29.59	31.46
	SSIM	0.892	0.858	0.915	0.953	0.884	0.908	0.878	0.888	0.890	0.874	0.894
ADMM.	PSNR	34.12	33.62	35.04	41.15	31.82	32.54	32.42	30.74	33.75	30.68	33.58
	SSIM	0.918	0.902	0.931	0.966	0.922	0.924	0.896	0.907	0.915	0.895	0.918
GAP-Net	PSNR	33.62	30.08	33.07	40.94	30.77	33.60	27.41	31.25	33.56	30.36	33.58
	SSIM	0.926	0.914	0.944	0.966	0.925	0.936	0.915	0.18	0.937	0.914	0.929
DGSMP	PSNR	33.26	32.09	33.06	40.54	28.86	33.08	30.74	31.55	31.66	31.44	32.63
	SSIM	0.915	0.898	0.925	0.964	0.882	0.937	0.886	0.923	0.911	0.925	0.917
MADNU	PSNR	34.73	35.32	35.10	40.17	32.32	34.42	33.09	32.62	34.75	32.17	34.46
	SSIM	0.918	0.903	0.927	0.961	0.898	0.912	0.887	0.933	0.924	0.931	0.919
HDNet	PSNR	34.95	32.52	34.52	43.00	32.49	**35.96**	29.18	34.00	34.56	32.22	34.34
	SSIM	0.948	0.953	0.957	0.981	0.957	0.965	0.937	0.961	0.958	0.950	0.957
HDN-SINR	PSNR	35.08	32.85	35.06	**43.21**	32.69	36.01	29.31	34.09	35.06	32.16	34.55
	SSIM	0.949	0.956	0.963	**0.985**	0.958	0.966	0.942	0.963	0.959	0.950	0.959
MAPUN	PSNR	35.11	36.11	36.40	41.92	32.78	35.07	33.98	33.03	35.81	32.81	35.30
	SSIM	0.945	0.950	0.953	0.973	0.952	0.959	0.931	0.956	0.957	0.946	0.952
MST-L	PSNR	35.40	35.87	36.51	42.27	32.77	34.80	33.66	32.67	35.39	32.50	35.18
	SSIM	0.941	0.944	0.953	0.973	0.947	0.955	0.925	0.948	0.949	0.941	0.948
CST-L	PSNR	35.96	36.84	**38.14**	42.44	33.25	35.72	34.86	**34.34**	36.51	33.09	36.12
	SSIM	0.949	0.955	0.962	0.975	0.955	0.963	0.944	0.961	0.957	0.945	0.957
EDUNet	PSNR	**36.48**	**37.65**	37.19	42.85	**34.29**	35.70	**35.37**	34.18	**36.81**	**33.46**	**36.40**
	SSIM	**0.951**	**0.961**	**0.963**	0.981	**0.962**	**0.966**	**0.949**	**0.962**	**0.960**	**0.951**	**0.961**

Table 5: Quantitative results in SAM/ERGAS on KAIST dataset in a noiseless setting. Best results are boldfaced.

Metric	λ-Net	TSA-Net	DGSMP	HDNet	MAPUN	MST-L	CST-L	EDUNet
SAM	19.71	8.75	8.94	6.68	6.19	7.47	5.81	**5.42**
ERGAS	108.63	90.78	31.50	25.64	24.56	27.72	22.92	**10.07**

Table 6: Comparisons on parameter numbers, FLOPs and time on KAIST dataset in a noiseless setting.

	λ-Net	ADMMNet	DNU	PnP-DIP	TSA-Net	DGSMP
#Param.	62.64M	4.27M	1.19M	33.85M	44.25M	3.76M
#FLOPs	117.98G	78.58G	163.48G	64.42G	110.06G	646.65G
Time	0.01s	0.70s	1.64s	9.79h	0.16s	1.39s
	MADNU	HDNet	MAPUN	MST-L	CST-L	EDUNet
#Param.	2.58M	2.35M	3.01M	2.03M	3.00M	1.51M
#FLOPs	134.17G	154G	117.9G	28.15G	40.10G	24.24G
Time	0.26s	0.02s	0.24s	0.41s	1.05s	0.15s

Table 7: PSNR(dB) results (mean±std.) on KAIST dataset with shot noise, obtained via 50 runs. Best results are boldfaced.

Noise	λ-Net	TSA-Net	DGSMP	HDNet	MST-L	CST-L	EDUNet
10bit	27.01±0.07	28.02±0.03	29.30±0.08	31.02±0.05	31.12±0.04	32.09±0.04	**32.23±0.04**
11bit	27.36±0.05	28.34±0.03	29.89±0.07	31.09±0.03	31.53±0.03	32.41±0.04	**32.80±0.03**
12bit	27.56±0.06	28.58±0.02	29.16±0.05	31.21±0.03	31.77±0.03	32.76±0.03	**33.02±0.02**

formance gains of EDUNet over other DUNs have again demonstrated the effectiveness of our NN architecture.

4.1.3. Visual Inspection

See Figure 3 for the visualization of HSI reconstruction results on two samples from the KAIST and Harvard datasets respectively. The spectral curves (density versus wavelength) correspond to the points marked by green boxes in the RGB references. In the legends of both figures, we provide the curve correlation value between the result of a compared method and the ground truth. Those values show that the HSIs reconstructed by the proposed EDUNet have the highest correlation to the ground truths. We also visualize three spectral channels of an entire reconstructed HSI and zoom in on the selected regions marked by yellow boxes. It can be seen that the results of

Table 8: Quantitative results on ICVL and Harvard datasets. Best results are boldfaced.

	Metric	SSNR	HSCNN	λ-Net	DNU	ADLTR	DTLP	HDNet	CST-L	EDUNet
ICVL	PSNR	30.40	28.45	29.01	32.61	-	34.53	36.38	36.23	**37.60**
	SSIM	0.943	0.934	0.946	0.966	-	0.977	0.981	0.981	**0.985**
	SAM	1.83	1.83	2.52	2.21	-	1.72	1.07	1.12	**0.69**
	ERGAS	50.60	61.57	50.70	37.53	-	30.07	9.37	6.09	**5.54**
Harvard	PSNR	31.14	27.60	29.37	31.11	31.67	32.43	34.26	34.50	**34.97**
	SSIM	0.942	0.895	0.909	0.929	0.935	0.941	0.948	0.952	**0.956**
	SAM	4.58	6.24	7.62	5.78	5.31	5.16	4.67	3.16	**2.53**
	ERGAS	74.91	105.89	62.51	73.53	71.08	62.51	32.45	27.89	**25.21**

EDUNet are more visually pleasing than that of other compared methods, with a better reconstruction of structures.

4.1.4. Further Analysis and Summary

Overall, in terms of average performance metrics, our EDUNet shows a notable improvement over prior models. While EDUNet delivers the best average reconstruction performance on KAIST, as shown in Table 4, there are instances where individual test samples exhibit performance slightly below the top values in terms of PSNR. This variance can largely be attributed to two factors: (a) the training samples not providing a sufficient coverage of the full characteristics of the test images; and (b) the various characteristics and out-of-distribution samples of test data. As a result, the observed performance variations of our EDUNet and other compared models can be attributed to the inherent diversity of dataset characteristics and the adaptivity capacity of each method to different image features/patterns. This phenomenon is not exclusive to our approach; similar performance variations have been reported in experiments with other HSI reconstruction methods such as [42, 44, 53]. These variations across different instances suggest that each method is particularly adept within a certain range of image patterns. Our experiments demonstrate that our EDUNet performs exceptionally well in the majority of instances, showcasing the widest adaptability range to image features and patterns.

Nevertheless, the instances where EDUNet is outperformed by other models provide valuable insights into potential areas for further refinement and optimization of our approach. In Table 4, the least favorable outcome for EDUNet occurs for the third instance; see Figure 4 for a visualization of re-

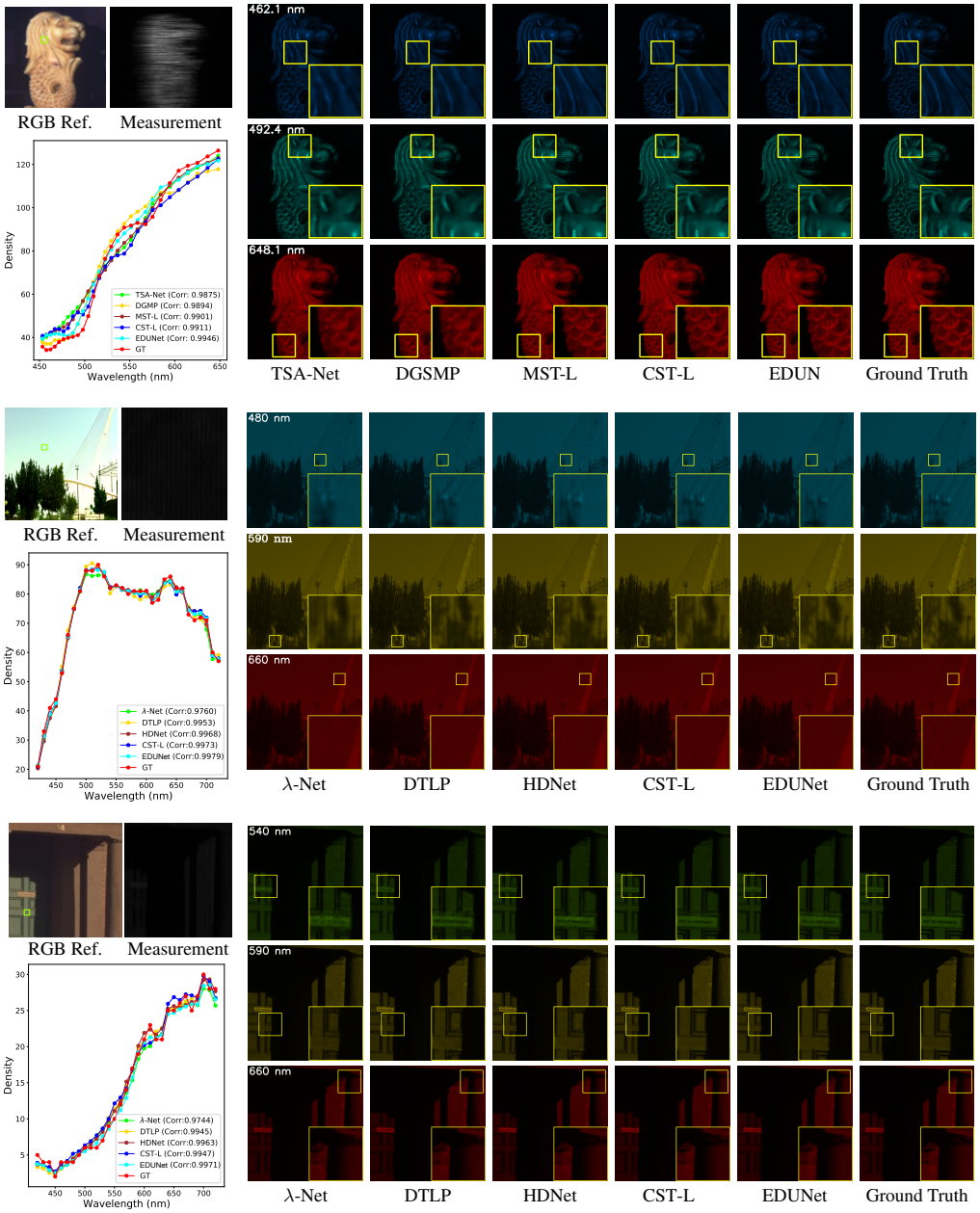

Figure 3: Visual comparison of HSI reconstruction on three samples from KAIST, ICVL and Harvard datasets respectively. Left: spectra curves of the selected regions marked by green boxes. Right: reconstruction on the spectral channels.

construction results of EDUNet and its top competitor, CST-L. We observe that this instance contains more flat black regions than other instances. It is likely that the cross-stage SA employed in EDUNet cannot fully utilize its advantages, leading to over-smoothed result with lower PSNR, likely due to involving information from the flat black regions to reconstruct the HSI. It is noteworthy that, despite achieving a higher PSNR value, CST-L introduces numerous artifacts. Consequently, our EDUNet achieves a higher SSIM value than CST-L, indicating better structural integrity in the reconstruction.

In terms of the other two metrics, SAM and ERGAS, EDUNet performs the best in all cases in Table 5. In addition, EDUNet is the only one that achieves the best performance in all metrics on the ICVL and Harvard datasets, as shown in Table 8. These results demonstrate the superiority of EDUNet in performance. Furthermore, Table 6 highlights that the EDUNet demonstrates a relatively-low model complexity in terms of the number of parameters, in comparison to existing methods. This aspect, coupled with the superior overall performance of EDUNet, suggests that our technique strikes an effective balance between accuracy and efficiency, both of which are important for practical applications.

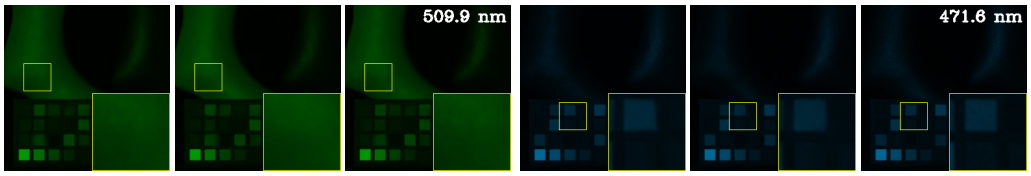

Figure 4: Visual comparison of reconstruction results on instance #3 of KAIST dataset.

4.2. Evaluation on Real Data

We also conduct a test on the real snapshots of spatial size 660×714 from [39, 41], which are captured by a real system with 28 wavelengths ranging from 450nm to 650nm and with 54-pixel dispersion in the column dimension. Following [39, 41], we use the mask associated with that real system to generate snapshots on both the CAVE and KAIST datasets, and then we inject 11-bit shot noise to the snapshots for simulating real situations. The resulting data is used to retrain our model. Due to the lack of ground truths in test data, we only compare the qualitative results of different methods. See Figure 5 for the reconstruction results on some real scenes.

The performance of EDUNet is also good on the real data. This indeed has demonstrated the good generalization performance of our model.

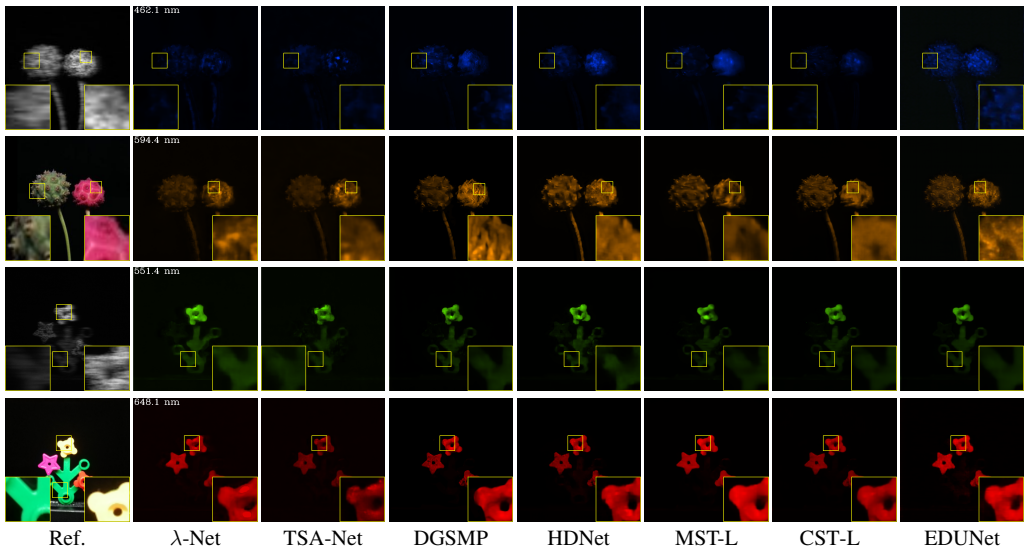

Figure 5: Visual comparison of HSI reconstruction on real data, in terms of two spectral channels.

4.3. Ablation Studies

Ablation studies are conducted on the KAIST dataset. We form some baselines by removing one or more main components of our approach. Concretely, we consider (a) "MAD→GD": replacing the MAD blocks by the GD steps in Eq. (4); (b) "MAD→Conv": replacing the MAD blocks with three convolutional layers, with $u^{(k)}$ and $x^{(k-1)}$ concatenated as the input. (c) "SA→ Conv": replacing the cross-stage SA with the same number of convolutional layers without cross-stage connections; (d) "Inner SA → Cross SA: replacing the cross-stage SA in the CAP network with the inner-stage SA which uses the features at current stage to calculate $\boldsymbol{K}^{(k)}, \boldsymbol{Q}^{(k)}, \boldsymbol{V}^{(k)}$ in (15); (e) "w/o $\mathcal{L}_{\text{sgc}}$" removing the SGC loss $\mathcal{L}_{\text{sgc}}$. For fair comparison, each baseline is configured to have (nearly) the same number of parameters as the original model, by uniformly increasing channel numbers of convolutional layers.

The results are listed in Table 9. To facilitate the comparison, we also include the competitive baseline, CST-L, in the table. It can be seen that each main component in our approach plays an important role. (a) Using

the MAD blocks as an alternate to GD steps can improve PSNR by almost 1db. (b) When implementing the learnable descent updating steps using plain convolutional layers instead of memory units, the performance will have around 0.92dB PSNR. This is probably because unlike our MAD blocks, convolutional layers cannot exploit the long-term dependencies among the descent updating steps. (c) When the whole cross-stage SA is disabled, further performance decrease is observed.(d) Benefiting from the power of SA, the cross-stage SA brings noticeable PSNR gain. The SA utilized in the cross-stage manner leads to around 0.44dB improvement in PSNR over that utilized in the inner-stage manner. (e) The SGC loss also has a solid contribution to the performance. See Figure 6 for an illustration of the effect of the SGC loss, where training with $\mathcal{L}_{\text{sgc}}$ makes the tendency of the predicted spectral curves closer to ground truths. We would like to mention that the triplet attention used in the CAP sub-networks also brings some PSNR gain (around 0.13dB). Its results are not listed in Table 9 as it is not a main contribution to this work.

Table 9: Results in ablation studies on KAIST dataset.

Metric	MAD →GD	MAD →Conv	SA →Conv	Cross SA →Inner SA	w/o $\mathcal{L}_{\text{sgc}}$	Original EDUNet	CST-L
PSNR(dB)	35.35/1.05↓	35.48/.92↓	35.81/.59↓	36.16/.44↓	35.98/.42↓	**36.40**	36.12
SSIM	0.947/.014↓	0.949/.012↓	0.956/.005↓	0.959/.002↓	0.958/.003↓	**0.961**	0.957
Time(s)	0.33/.18↑	0.38/.13↑	0.43/.06↑	0.51/.00	0.51/.00	0.51	1.05

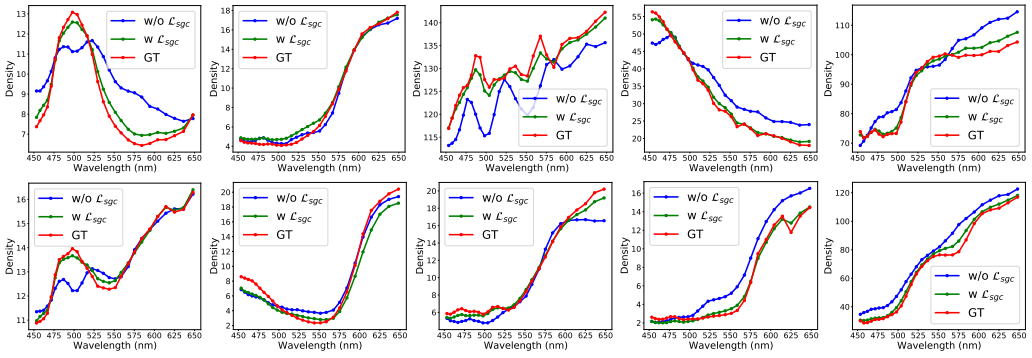

Figure 6: Spectra of selected regions on Scene#1, Scene#2, Scene#6, Scene#7 and Scene#10 (From left to right) of KAIST dataset.

To evaluate the influence of stage number K in EDUNet, we vary it

from 1 to 7 and then test the performance. Note that when $K = 1$ there is no cross-stage link. In this case, we replace the cross-stage SA with the standard SA and the ConvLSTM with three convolutional layers of a similar size. The PSNR results on KAIST dataset are listed in Table 10. It can be seen that, as the stage number K increases, the reconstruction performance of EDUNet is continuously improved, and meanwhile the model size of EDUN is also increased. The increase of model size is not significant which suggests that EDUN is computationally scalable to the stage number. To balance both performance and computational complexity, we take $K = 6$ in the previous experiments, and using $K = 7$ may lead to further improvement. In addition, we also constructed two additional baselines by reducing the stage number to $K = 2, 4$ respectively, while increasing the channel number to keep the model size close to the original one. The PSNR results (not listed in Table 10 to avoid confusion) are 32.56dB ($K = 2$) and 34.17dB ($K = 4$) respectively, in comparison to the original one 36.40dB ($K = 6$). Such a PSNR decrease along with the decrease of K implies the memory-based update steps potentially benefit more from larger stage numbers, *i.e.*, the MAD blocks can well exploit the long-term dependencies among different stages to bring improvement for DUNs.

Table 10: Average PSNR on KAIST dataset and parameter numbers *w.r.t.* different stage numbers.

Metric	$K = 1$	$K = 2$	$K = 3$	$K = 4$	$K = 5$	$K = 6$	$K = 7$
PSNR(dB)	28.74	31.48	32.83	33.89	35.37	36.40	37.03
#Param.(M)	0.25	0.50	0.76	1.00	1.26	1.51	1.76

To better compare the performance of our proposed EDUNet with that of other DUNs, we plot the PSNR-GFLOPs curves *w.r.t.* the stage number of the DUN; see Fig. 7. We can see that with the increase of stage numbers, the PSNR increment of EDUNet is faster than that of other DUNs, and the overall performance and computational complexity of EDUNet is better than that other DNUs. Such advantages come from the proposed memory-assistant descent updating steps and the cross-stage SA modules.

5. Conclusion

In this work, we investigated enhancements of DUNs for CASSI-based HSI reconstruction. Our proposed model, EDUNet, incorporates three key aug-

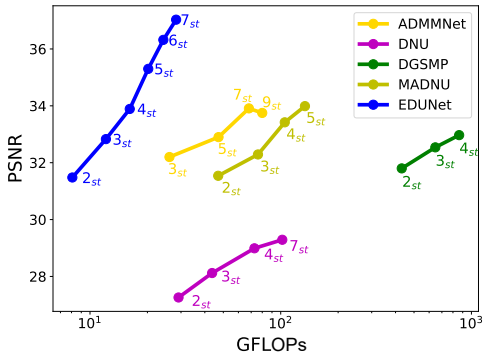

Figure 7: PSNR-GFLOPs curves *w.r.t.* stage of number of a DUN. Each curve is for a DUN. The sequential numbers under each curve denote the corresponding stage number.

mentations: ConvLSTM-assistant gradient-driven update steps inspired by momentum-based acceleration, cross-stage self-attentive NNs for efficiently exploring inherent self-similarities in HSIs, and a spectral geometry consistency loss function for better regularization. The integration of these enhancements resulted in a light-weight model that exhibited a noticeable improvement in reconstruction accuracy over state-of-the-art methods. This improvement was demonstrated through extensive experiments on three benchmark datasets using four quantitative metrics, as well as on several real samples using visual inspection.

The findings of this paper directly contribute to the field by addressing the practical challenge of reconstructing high-quality HSIs from compressed data, a crucial step for the various applications of hyperspectral imaging. This advancement not only enriches the technical aspects of hyperspectral imaging but also significantly broadens its practical applicability in the industry. While our approach is tailored for CASSI-based HSI reconstruction, the underlying concept holds potential for applications in other compressive imaging problems. We will explore these possibilities in our future research.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China under Grant 62372186, in part by Natural Science Foundation of Guangdong Province under Grants 2022A1515011755 and 2023A1515012841, in part by Fundamental Research Funds for the Central Universities under Grant x2jsD2230220, and in part by Singapore MOE AcRF Tier 1 under

Grant A-8000981-00-00.

References

[1] S. L. Ustin, J. A. Gamon, Remote sensing of plant functional types, New Phytologist 186 (4) (2010) 795–816.

[2] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, J. Chanussot, Hyperspectral remote sensing data analysis and future challenges, IEEE Geoscience and remote sensing magazine 1 (2) (2013) 6–36.

[3] C. A. Bishop, J. G. Liu, P. J. Mason, Hyperspectral remote sensing for mineral exploration in pulang, yunnan province, china, International Journal of Remote Sensing 32 (9) (2011) 2409–2426.

[4] T. Vo-Dinh, A hyperspectral imaging system for in vivo optical diagnostics, IEEE Engineering in Medicine and Biology Magazine 23 (5) (2004) 40–49.

[5] M. B. Stuart, A. J. McGonigle, J. R. Willmott, Hyperspectral imaging in environmental monitoring: A review of recent developments and technological advances in compact field deployable systems, Sensors 19 (14) (2019) 3071.

[6] A. Wagadarikar, R. John, R. Willett, D. Brady, Single disperser design for coded aperture snapshot spectral imaging, Applied Optics 47 (10) (2008) B44–B51.

[7] Z. Xiong, Z. Shi, H. Li, L. Wang, D. Liu, F. Wu, Hscnn: Cnn-based hyperspectral image recovery from spectrally undersampled projections, in: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshop, 2017, pp. 518–525.

[8] L. Wang, C. Sun, Y. Fu, M. H. Kim, H. Huang, Hyperspectral image reconstruction using a deep spatial-spectral prior, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8032–8041.

[9] X. Miao, X. Yuan, Y. Pu, V. Athitsos, λ-net: Reconstruct hyperspectral images from a snapshot measurement, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4059–4069.

[10] L. Wang, C. Sun, M. Zhang, Y. Fu, H. Huang, Dnu: deep non-local unrolling for computational spectral imaging, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1661–1671.

[11] Y. Fu, T. Zhang, L. Wang, H. Huang, Coded hyperspectral image reconstruction using deep external and internal learning, IEEE Transactions on Pattern Analysis and Machine Intelligence (2021).

[12] X. Zhang, Y. Zhang, R. Xiong, Q. Sun, J. Zhang, Herosnet: Hyperspectral explicable reconstruction and optimal sampling deep network for snapshot compressive imaging, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2022).

[13] P. L. Combettes, J.-C. Pesquet, Proximal splitting methods in signal processing, in: Fixed-point algorithms for inverse problems in science and engineering, Springer, 2011, pp. 185–212.

[14] A. Nitanda, Stochastic proximal gradient descent with acceleration techniques, Advances in Neural Information Processing Systems 27 (2014).

[15] R. Ma, J. Miao, L. Niu, Transformed l1 regularization for learning sparse deep neural networks, Neural Networks 119 (08 2019). doi:10.1016/j.neunet.2019.08.015.

[16] Y. Yang, R. Tao, K. Wei, Y. Fu, Dynamic proximal unrolling network for compressive imaging, Neurocomputing 510 (2022) 203–217.

[17] T. Tieleman, G. Hinton, et al., Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, COURSERA: Neural networks for machine learning 4 (2) (2012) 26–31.

[18] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[19] Y. Liang, J. Liu, D. Xu, Stochastic momentum methods for non-convex learning without bounded assumptions, Neural Networks (2023).

[20] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, W.-c. Woo, Convolutional lstm network: A machine learning approach for precipitation nowcasting, Advances in Neural Information Processing Systems 28 (2015).

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in Neural Information Processing Systems 30 (2017).

[22] I. Choi, D. S. Jeon, G. Nam, D. Gutierrez, M. H. Kim, High-quality hyperspectral reconstruction using a spectral prior, ACM Transactions on Graphics 36 (6) (2017) 218:1–13. doi:10.1145/3130800.3130810.
URL http://dx.doi.org/10.1145/3130800.3130810

[23] Y. Fu, Y. Zheng, I. Sato, Y. Sato, Exploiting spectral-spatial correlation for coded hyperspectral image restoration, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2016, pp. 3727–3736.

[24] Y. Liu, X. Yuan, J. Suo, D. J. Brady, Q. Dai, Rank minimization for snapshot compressive imaging, IEEE Transactions on Pattern Analysis and Machine Intelligence 41 (12) (2018) 2990–3006.

[25] S. Zhang, L. Wang, Y. Fu, X. Zhong, H. Huang, Computational hyperspectral imaging based on dimension-discriminative low-rank tensor recovery, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 10183–10192.

[26] Y. Chen, T.-Z. Huang, W. He, N. Yokoya, X.-L. Zhao, Hyperspectral image compressive sensing reconstruction using subspace-based nonlocal tensor ring decomposition, IEEE Transactions on Image Processing 29 (2020) 6813–6828.

[27] L. Wang, S. Zhang, H. Huang, Adaptive dimension-discriminative low-rank tensor recovery for computational hyperspectral imaging, International Journal of Computer Vision 129 (10) (2021) 2907–2926.

[28] X. Yuan, Y. Liu, J. Suo, Q. Dai, Plug-and-play algorithms for large-scale snapshot compressive imaging, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 1447–1457.

[29] H. Qiu, Y. Wang, D. Meng, Effective snapshot compressive-spectral imaging via deep denoising and total variation priors, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 9127–9136.

[30] Z. Meng, Z. Yu, K. Xu, X. Yuan, Self-supervised neural networks for spectral snapshot compressive imaging, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021).

[31] Y. Quan, X. Qin, M. Chen, Y. Huang, High-quality self-supervised snapshot hyperspectral imaging, in: ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2022, pp. 1526–1530.

[32] L. Wang, T. Zhang, Y. Fu, H. Huang, Hyperreconnet: Joint coded aperture optimization and image reconstruction for compressive hyperspectral imaging, IEEE Transactions on Image Processing 28 (5) (2018) 2257–2270.

[33] T. Zhang, Y. Fu, L. Wang, H. Huang, Hyperspectral image reconstruction using deep external and internal learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8559–8568.

[34] K. Yorimoto, X.-H. Han, Hypermixnet: Hyperspectral image reconstruction with deep mixed network fryom a snapshot measurement, in: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshop, 2021, pp. 1184–1193.

[35] X. Hu, Y. Cai, J. Lin, H. Wang, X. Yuan, Y. Zhang, R. Timofte, L. Van Gool, Hdnet: High-resolution dual-domain learning for spectral compressive imaging, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2022).
URL https://doi.org/10.1109/CVPR52688.2022.01702

[36] Y. Cai, J. Lin, X. Hu, H. Wang, X. Yuan, Y. Zhang, R. Timofte, L. Van Gool, Mask-guided spectral-wise transformer for efficient hyperspectral image reconstruction, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2022).

[37] J. Ma, X.-Y. Liu, Z. Shou, X. Yuan, Deep tensor admm-net for snapshot compressive imaging, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 10223–10232.

[38] Z. Meng, X. Yuan, S. Jalali, Deep unfolding for snapshot compressive imaging, International Journal of Computer Vision (2023) 1–26.

[39] T. Huang, W. Dong, X. Yuan, J. Wu, G. Shi, Deep gaussian scale mixture prior for spectral compressive imaging, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 16216–16225.

[40] C. Mou, Q. Wang, J. Zhang, Deep generalized unfolding networks for image restoration, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2022).

[41] Z. Meng, J. Ma, X. Yuan, End-to-end low cost compressive spectral imaging with spatial-spectral self-attention, in: European Conference on Computer Vision, Springer, 2020, pp. 187–204.

[42] Y. Cai, J. Lin, X. Hu, H. Wang, X. Yuan, Y. Zhang, R. Timofte, L. V. Gool, Coarse-to-fine sparse transformer for hyperspectral image reconstruction, in: European Conference on Computer Vision, 2022.

[43] J. Song, B. Chen, J. Zhang, Memory-augmented deep unfolding network for compressive sensing, in: Proceedings of the ACM International Conference on Multimedia, 2021.

[44] J. Song, B. Chen, J. Zhang, Deep memory-augmented proximal unrolling network for compressive sensing, International Journal of Computer Vision 131 (6) (2023) 1477–1496.

[45] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[46] M. Zhou, K. Yan, J. Pan, W. Ren, Q. Xie, X. Cao, Memory-augmented deep unfolding network for guided image super-resolution, International Journal of Computer Vision (2022).

[47] P. Tseng, On accelerated proximal gradient methods for convex-concave optimization, SIAM Journal on Optimization 2 (3) (2008).

[48] D. Misra, T. Nalamada, A. U. Arasanipalai, Q. Hou, Rotate to attend: Convolutional triplet attention module, in: Proceedings of the IEEE/CVF winter conference on applications of computer vision, 2021, pp. 3139–3148.

[49] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., Pytorch: An imperative style, high-performance deep learning library, Advances in Neural Information Processing Systems 32 (2019).

[50] S. Zheng, Y. Liu, Z. Meng, M. Qiao, Z. Tong, X. Yang, S. Han, X. Yuan, Deep plug-and-play priors for spectral snapshot compressive imaging, Photonics Research 9 (2) (2021) B18–B29.

[51] X. Wang, J. Ma, J. Jiang, X.-P. Zhang, Dilated projection correction network based on autoencoder for hyperspectral image super-resolution, Neural Networks 146 (2022) 107–119.

[52] F. Yasuma, T. Mitsunaga, D. Iso, S. K. Nayar, Generalized assorted pixel camera: postcapture control of resolution, dynamic range, and spectrum, IEEE Transactions on Image Processing 19 (9) (2010) 2241–2253.

[53] H. Chen, W. Zhao, T. Xu, G. Shi, S. Zhou, P. Liu, J. Li, Spectral-wise implicit neural representation for hyperspectral image reconstruction, IEEE Transactions on Circuits and Systems for Video Technology (2023).

[54] B. Arad, O. Ben-Shahar, Sparse recovery of hyperspectral signal from natural rgb images, in: European Conference on Computer Vision, Springer, 2016, pp. 19–34.

[55] A. Chakrabarti, T. Zickler, Statistics of real-world hyperspectral images, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2011, pp. 193–200.

[56] S. Zhang, L. Wang, L. Zhang, H. Huang, Learning tensor low-rank prior for hyperspectral image reconstruction, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 12006–12015.