Neurocomputing 629 (2025) 129723

Contents lists available at ScienceDirect

Neurocomputing

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

PLV-CSNet: Projected Landweber Variant unfolding network for image compressive sensing reconstruction ☆

Junpeng Hao [a,1], Huang Bai [a,1], Xiumei Li [a,*], Marko Panic [b], Junmei Sun [a]

[a] School of Information Science and Technology, Hangzhou Normal University, 311121, Hangzhou, China
[b] BioSense Institute, University of Novi Sad, 21000 Novi Sad, Serbia

ARTICLE INFO

Communicated by R. Yang

Keywords:
Compressive sensing
Deep unfolding
Projected Landweber
Measurement residual

ABSTRACT

Due to the powerful learning capability and fast processing speed of deep neural networks, a series of pure data-driven and deep unfolding networks for image reconstruction have emerged, achieving improved reconstruction quality. These reconstruction networks typically employ convolutional neural networks or residual neural networks to extract high-dimensional features of the dominant structure component. However, the edge and texture components in multi-dimensional features as well as the measurement residual generated at each iteration during the unfolding procedure are often neglected, which would affect the quality of image reconstruction. In this paper, a projected Landweber variant unfolding network (PLV-CSNet) is proposed for image compressive sensing reconstruction. A PLV algorithm is investigated and then unfolded to the PLV-Block, which consists of a thresholding module (TSM) and a progressive projecting module (PPM). The TSM utilizes the dense block to fuse multi-dimensional image features and the soft thresholding to eliminate image noise. The PPM combines the approximate message passing algorithm with deep neural networks to compute the projections of the approximation solution for images, as well as calculate the measurement residual generated during each iteration. Furthermore, a residual integration module (RIM) is designed to employ the measurement residuals to reconstruct the image residual which are then flexibly supplemented back into the reconstructed image. The effectiveness of PLV-CSNet is demonstrated in four standard benchmark datasets, and comparisons with classical image compressive sensing reconstruction networks show that our network could achieve higher reconstruction accuracy. Codes are available at: https://github.com/junp-hao/PLV-CSNet.

1. Introduction

Compressive sensing (CS) is a signal processing theory that provides a more efficient way to acquire and reconstruct signals [1,2]. In the CS framework, when signals are sparse in certain transform domains, they can be reconstructed through optimization algorithms from fewer samples than that required by the Nyquist theorem, thereby alleviating the burden on data storage and transmission bandwidth. CS has been applied in many practical applications [3–5], including single-pixel imaging, fast magnetic resonance imaging, snapshot compressive imaging, and video CS.

The sampling process of CS can be represented as a linear mapping $y = \Phi x$, where $x \in \mathbb{R}^{N\times 1}$ is the original signal, $y \in \mathbb{R}^{M\times 1}$ is the measurement, $\Phi \in \mathbb{R}^{M\times N}$ denotes the sampling matrix, and $\frac{M}{N}$ represents the CS ratio. If Φ adheres to the restricted isometry property, the original signal can be reconstructed with high probability. As $M \ll N$, recovering the original signal from the measurement is a typical NP-hard ill-posed problem. Traditional CS optimization algorithms, including convex relaxation algorithms [6], greedy algorithms [7,8], and Bayesian algorithms [9], are usually based on prior knowledge on the sparsity of signals in specific domains. Although these methods often have theoretical guarantees and interpretability, employing them to reconstruct images typically incurs high computational costs, and the quality of reconstructed images still needs to be improved.

In recent years, deep learning (DL) has been widely applied in computer vision with superior performance [10–13]. Due to the robust learning and representation capabilities of neural networks [14,15], various pure data-driven networks have been proposed for image CS

☆ This work was supported in part by the Natural Science Foundation of China under Grants 61571174 and 61801159, and in part by the China-Croatia Bilateral Science & Technology Cooperation Project.

* Corresponding author.
E-mail addresses: junp@stu.hznu.edu.cn (J. Hao), baihuang@hznu.edu.cn (H. Bai), lixiumei@hznu.edu.cn (X. Li), panic@biosense.rs (M. Panic), junmeisun@hznu.edu.cn (J. Sun).

[1] Junpeng Hao and Huang Bai contribute equally and are co-first authors.

https://doi.org/10.1016/j.neucom.2025.129723
Received 25 November 2024; Received in revised form 13 January 2025; Accepted 8 February 2025
Available online 17 February 2025

reconstruction, aiming to solve the reconstruction problem by directly learning the inverse mapping from measurement y to the original signal x [16–19]. Convolutional neural networks (CNNs) and residual neural networks (ResNet) in DL can accurately capture the structural features of real images through abundant training samples, significantly enhancing image reconstruction precision. DL can transform the separated designs of sampling and reconstruction processes in the traditional CS model into an end-to-end pure data-driven architecture, and the conventional linear Gaussian random sampling matrix can be converted into an adaptive sampling network. However, the black box characteristic of pure data-driven networks leads to a lack of mathematical interpretability [20]. The construction of network lacks theoretical guidance, and the robustness of the network is difficult to be guaranteed.

Combining the theoretical advantages of traditional optimization algorithms with the high performance of DL, deep unfolding networks adopt traditional CS optimization algorithms to guide the design of neural networks. In deep unfolding networks, each iteration of the traditional optimization algorithm is unfolded into a layer of the network, and the parameters involved in the algorithm are transformed into network parameters. Consequently, the network's forward propagation is similar to executing a limited number of progressive iterations of the traditional algorithm. The training of a deep unfolding network involves an iterative algorithmic process guided by prior knowledge, which enhances the network's mathematical interpretability.

Many deep unfolding CS networks have been successfully applied to improve image reconstruction quality and ensure network interpretability, by utilizing CNN or ResNet [21] to extract high-dimensional image features of the dominant structure component to reconstruct the image and alleviate gradient vanishing. However, CNN or ResNet usually fail to fully exploit other multi-dimensional features in images, such as edge and texture details. First, traditional CNN has a limited receptive field, focusing primarily on local spatial information, which makes it difficult to capture long-range dependencies and complex cross-dimensional relationships. ResNet does not efficiently facilitate the reuse or fusion of features across layers, which can result in some low-dimensional features being lost in the deeper layers. To overcome these limitations, the proposed network utilizes the DenseNet architecture, which effectively addresses the shortcomings of CNN and ResNet in exploiting multi-dimensional features. In DenseBlock (DB) of DenseNet, each layer is connected to all preceding layers. This dense connectivity not only improves gradient flow and learning efficiency but also facilitates feature reuse, allowing low-dimensional features to be combined with high-dimensional representations.

Furthermore, the measurement residual generated by each iteration step are often ignored and the rich information in the measurement is not fully utilized in existing deep unfolding CS networks. Therefore, to achieve a better image reconstruction performance, it is necessary to supplement the measurement residuals into the reconstructed image. Firstly, the reconstructed image in each iteration step of the deep reconstruction subnet is sampled to generate the measurement. Then the measurement is subtracted from the measurement in preliminary reconstruction subnet to obtain the measurement residuals which are employed to reconstruct the image residuals.

Different deep unfolding networks unfold various traditional optimization algorithms, whereas the proposed network specifically unfolds a variant of the PL algorithm. Specifically, the projection operation in PL is replaced with the approximate message passing (AMP) algorithm, and the soft thresholding is utilized instead of the hard thresholding used in PL. Moreover, measurement residuals are incorporated into the iterative process to effectively mitigate the loss of important image details. Finally, the DB is employed for multi-dimensional feature extraction, enabling the network to preserve complex data features and capture intricate cross-dimensional dependencies.

This paper proposes a Projected Landweber [22] Variant unfolding network for image CS reconstruction (PLV-CSNet) based on a novel PLV algorithm. Designing a specialized PLV-Block, each iteration of the PLV algorithm can be split into two steps corresponding to a thresholding module (TSM) and a progressive projecting module (PPM). The TSM is designed for multi-dimensional feature fusion and thresholding operations. The PPM computes the projections of the approximation solution for images and calculates measurement residuals by repeatedly using the measurement. Finally, the residual integration module (RIM) reconstructs the measurement residuals into image residuals and flexibly supplements them into the reconstructed image to get the fine-tuned image.

The main contributions of this paper are as follows:

- A novel deep unfolding CS network PLV-CSNet is proposed based on the PLV algorithm. The iterative PLV-Block is constructed by unfolding the projection and thresholding operations, and each PLV-Block corresponds to one iteration. Extensive experiments in various datasets demonstrate that the proposed PLV-CSNet not only enhances image reconstruction quality but also reduces network parameters.
- The TSM is proposed to capture richer image details and to eliminate image noise. In the TSM, the DB is employed to extract multi-dimensional image features for utilization in the unfolding algorithm, and soft thresholding is subsequently utilized to eliminate image noise and improve the quality of image reconstruction.
- The PPM is introduced to progressively approximate the solution to the projection operation by unfolding AMP. Additionally, the PPM utilizes the measurement to calculate the measurement residual generated at each iteration.
- The RIM is designed to enhance reconstruction quality by reconstructing the measurement residuals to image residuals and then supplementing them into the reconstructed image. The RIM flexibly supplements various image residuals into the reconstructed image, resulting in a smoother result.

The remainder of this paper is structured as follows: In Section 2, we provide a brief review of related works and introduce the PL algorithm. Section 3 describes the architecture of PLV-CSNet in detail. In Section 4, we discuss the results of extensive experiments and provide further analysis. Finally, we conclude our work in Section 5.

2. Related works

This section briefly introduces traditional optimization algorithms especially the PL algorithm, pure data-driven networks, and deep unfolding networks.

2.1. Traditional optimization algorithms for CS

Traditional CS optimization algorithms address the L_0 norm sparsity regularization optimization problem under prior constraints. Typically, they transform the ill-posed problem into a convex optimization problem minimized by the L_1 norm or utilize iterative greedy algorithms to get the approximation solution progressively.

Under the condition of sparsity, addressing the signal inverse problem is mathematically modeled as:

$$\min_x \frac{1}{2}\|\Phi x - y\|_2^2 + \lambda\|\Psi x\|_1 \tag{1}$$

where λ represents the regularization parameter. The L_1 norm term constrains the sparsity of signal x in the transform domain Ψ.

Representative algorithms for solving optimization problems of Eq. (1) include the basis pursuit (BP) algorithm [23], interior-point method (IPM) [24], iterative shrinkage-thresholding algorithm (ISTA) [25], alternating direction method of multipliers (ADMM) [26], and approximate message passing (AMP) [27]. Moreover, some works have aimed to enhance the reconstruction performance by exploring image priors. Li et al. proposed TVAL3 by utilizing total variation regularization to enhance local smoothness for image reconstruction [28].

Metzler et al. combined the BM3D denoiser with the AMP algorithm and proposed a new reconstruction framework called D-AMP [29]. However, these methods lead to high computational costs and face challenges in prior design or parameter settings.

The PL algorithm [22] has attracted significant attention in the field of inverse problems, particularly in tasks related to image reconstruction. The PL algorithm is derived from the classical Landweber iteration method in optimization theory. During the iteration process, a projection operation is introduced to update the current estimation, thereby improving convergence feasibility and settlement. Additionally, a threshold operation is incorporated to achieve specific regularization effects, enhancing the sparsity of the solution. The PL algorithm can be expressed as follows:

$$\hat{x}^k = x^k + \frac{1}{\gamma}\boldsymbol{\Phi}^T(y - \boldsymbol{\Phi}x^k) \tag{2}$$

$$x^{k+1} = \begin{cases} \hat{x}^k, & |\hat{x}^k| \geq \tau^k \\ 0, & |\hat{x}^k| < \tau^k \end{cases} \tag{3}$$

where $\boldsymbol{\Phi}^T$ represents the transpose of the sampling matrix in CS, $\frac{1}{\gamma}$ is the largest eigenvalue of $\boldsymbol{\Phi}\boldsymbol{\Phi}^T$, and τ^k is a threshold which is set appropriately during each iteration.

Eqs. (2) and (3) involve a two-step process in the PL algorithm: the first step performs a projection operation to progressively obtain an approximate solution of the signal. In contrast, the second step applies hard thresholding in the PL algorithm to eliminate noise. CS reconstruction based on the PL algorithm not only reduces computational complexity but also, more importantly, offers the distinct advantage of readily incorporating additional optimization techniques. For example, Mun et al. combined Wiener filtering with the traditional PL iterative process, proposing the smooth projected Landweber (SPL) algorithm for image CS reconstruction [30]. Based on SPL algorithm, Fowler et al. [31] proposed variant algorithms to improve the reconstruction performance.

On one hand, the projection operation in the PL algorithm inherently faces issues such as lower image reconstruction quality and slower convergence speed. In contrast, the AMP algorithm can obtain an approximate solution of the original signal while reducing iteration costs and accelerating convergence, thereby enhancing reconstruction performance. On the other hand, hard thresholding leads to detail loss and artifacts, causing rough edges and reduced image quality. In contrast, soft thresholding removes noise while preserving low-amplitude signals, maintaining finer image details without introducing artifacts. Therefore, we replace the hard thresholding in the PL algorithm with soft thresholding. In the subsequent section, we present a PLV algorithm that integrates the AMP algorithm and the soft thresholding procedure within the framework of PL iterations. This variant is designed to yield reconstructed images that not only highly approximate the original image but also improve the computational efficiency.

2.2. Pure data-driven networks for CS

The learning capability of neural networks and their success in computer vision have led to the emergence of numerous pure data-driven networks for CS. In 2015, Mousavi et al. proposed stacked denoising autoencoders (SDA) as self-supervised feature learners to reconstruct image blocks using undersampled measurement [16]. In 2016, inspired by image super-resolution networks, Kulkarni et al. introduced ReconNet, which was the first use of CNN for non-iterative CS reconstruction [17]. In 2019, Du et al. [32] used a fully convolutional measurement network to learn the inverse mapping. Yao et al. introduced residual learning based on ReconNet, replacing convolutional layers with residual learning blocks, proposing DR^2-Net [33]. In 2020, Shi et al. proposed CSNet$^+$, an end-to-end image reconstruction neural network that simulated traditional sampling matrix using convolutional layers, achieving excellent reconstruction performance [18]. In 2020, Chen et al. proposed Mac-Net, a cascaded image CS reconstruction network that decomposed image reconstruction into a cascade of detail reconstruction modules and residual update modules, incorporating context memory modules for information interaction [19]. Furthermore, in 2023, Gan et al. introduced a novel hierarchical network, TCS-Net [34], which was a Transformer-based approach and was distinguished by its two-stage decoding process: a patch-wise decoding phase and a pixel-wise decoding phase. However, due to the black box characteristic of neural networks, pure data-driven methods are unable to explain the reconstruction process. Moreover, the training procedure requires a lot of data, and fine-tuning the parameters of the network is not straightforward since the number of parameters is huge.

2.3. Deep unfolding networks for CS

Recently, many deep unfolding networks that combine traditional optimization algorithms with DL have been proposed. In 2018, inspired by ISTA, Zhang et al. proposed ISTA-Net and ISTA-Net$^+$, transforming ISTA into a deep network for image CS reconstruction [35]. In 2019, Dong et al. proposed the DPDNN to solve common inverse problems by unfolding the denoising process which was inspired by half quadratic splitting (HQS) [36]. In 2020, Yang et al. unfolded the popular optimization algorithm ADMM into a deep unfolding network ADMM-CSNet, which was used to solve CS and related sparse constraint estimation problems [37]. Zhang et al. proposed an interpretable deep unfolding network OPINE-Net [38], which can be regarded as a variant of ISTA-Net$^+$ and simultaneously explored adaptive sampling and reconstructing. In 2021, You et al. proposed a new end-to-end flexible COAST to unfold ISTA, with superior performance and remarkable flexibility [39]. In 2021, Zhang et al. proposed AMP-Net, a new deep unfolding network based on the denoising perspective of the AMP algorithm [40]. In 2022, Shen et al. proposed a novel Transformer-based hybrid architecture TransCS to achieve high-quality image reconstruction [41]. In 2023, Ye et al. proposed a hybrid framework, CSformer [42], which integrated detailed spatial information from CNN and the global context provided by transformer. Song et al. proposed OCTUF [43] to utilize cross-attention mechanisms to optimize the image. In recent years, researchers have also proposed many improved deep unfolding networks for CS [44–50]. When using these deep unfolding networks to extract image features, edge and texture components in multi-dimensional features are often ignored, which will decrease the accuracy of the following image reconstruction.

In most of CS reconstruction networks, such as CSNet$^+$ [18] and DR^2-Net [33], residual blocks derived from ResNet [21] are commonly used in place of CNN to alleviate the gradient vanishing problem, achieving high-quality image reconstruction. Compared to ResNet, the DenseNet [51] proposes a more dense connectivity scheme, where every layer is interconnected with all previous layers. Specifically, the feature maps of all previous layers are used as additional inputs and are connected by cross-channel concatenation. Because each layer establishes connections with previous layers, error signals can easily propagate to previous layers, allowing these previous layers to receive direct supervision from the final feature layer. Due to the dense connections, DB can fully leverage hierarchical features from all convolutional layers, thereby obtaining richer image detail information and achieving feature reuse for high-quality image reconstruction. In addition, unfolding the traditional optimization algorithm usually generates certain image residuals during each iteration step. These issues may affect the quality of image reconstruction. Therefore, a novel deep unfolding network dubbed PLV-CSNet is proposed in this paper to tackle the above issues.

3. PLV-CSNet architecture

In Sections 3.1 and 3.2, the overview of the PLV algorithm and the PLV-CSNet architecture are introduced. The preliminary reconstruction subnet (PRS), the deep reconstruction subnet (DRS) and loss function are introduced in Sections 3.3, 3.4 and 3.5, respectively.

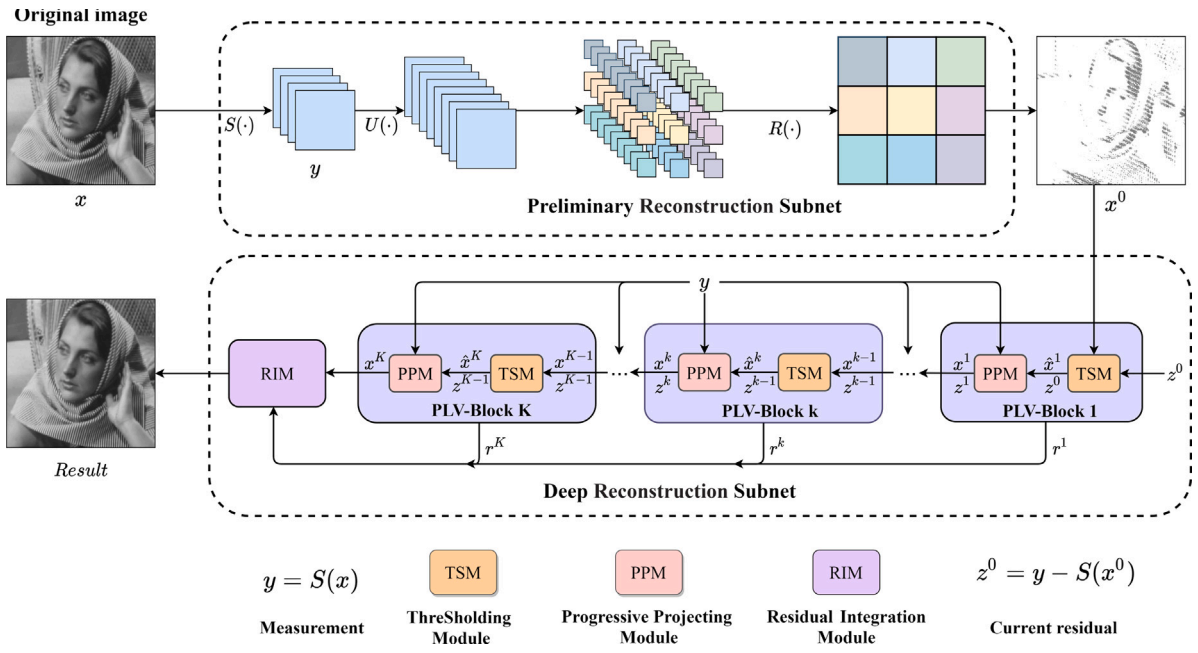


Fig. 1. The architecture of the PLV-CSNet.

3.1. PLV algorithm

The AMP algorithm can obtain the approximation solution of the original signal, reduce iteration costs, and improve convergence speed, thus enhancing reconstruction performance [27]. The specific iterative process of the AMP algorithm is represented as:

$$\begin{cases} \theta^k = \mu_k'(x^{k-1} + \boldsymbol{\Phi}^T z^{k-1}) \\ z^k = y - \boldsymbol{\Phi} x^k + \theta^k z^{k-1} \\ x^{k+1} = \mu_k(x^k + \boldsymbol{\Phi}^T z^k) \end{cases} \tag{4}$$

where θ^k is the scale factor of the Onsager reaction item to improve the phase transition, z^k is the current residual, $\mu_k(\cdot)$ represents the threshold shrinkage function, and $\mu_k'(t)$ is its derivative of function input with $\mu_k'(t) = \frac{\partial \mu_k(t)}{\partial t}$.

Soft thresholding is effective in removing noise from the image without completely eliminating low-amplitude signal components, which could preserve more subtle image details without introducing artifacts in the image [52]. the soft thresholding algorithm is used for denoising, which is denoted as:

$$soft(s_1^k, \tau^k) = \begin{cases} s_1^k + \tau^k, & s_1^k \le -\tau^k \\ 0, & |s_1^k| \le \tau^k \\ s_1^k - \tau^k, & s_1^k \ge \tau^k \end{cases} \tag{5}$$

where τ^k denotes the threshold parameter. In our network, the threshold parameter in soft thresholding serves as a learnable parameter that is continuously updated with the network to adapt to the noise level in the image.

The PLV algorithm integrates the AMP algorithm and the soft thresholding to replace the projection operation and hard thresholding within the PL algorithm iterations. Each iteration of the PLV algorithm can be split into two steps corresponding to the PPM and the TSM. The PPM computes the projections of the approximation solution for images and calculates measurement residuals by repeatedly using the measurement. The TSM is designed for multi-dimensional feature fusion and thresholding operations. More details about PLV-CSNet will be presented in the following.

3.2. Overview of PLV-CSNet architecture

As illustrated in Fig. 1, PLV-CSNet consists of one PRS and one DRS. The PRS consists of three steps: sampling $S(\cdot)$, upsampling $U(\cdot)$ and reshaping $R(\cdot)$. The $S(\cdot)$ represents the process of sampling the original image x, resulting in the output of the measurement y. Then, the y are upsampled by $U(\cdot)$. Finally, $R(\cdot)$ reshapes the upsampled values to generate the preliminary reconstructed image x^0. The DRS consists of a series of PLV-Blocks and one RIM, where each PLV-Block corresponds to one iteration of the PLV algorithm. The inputs to the DRS include three components: y from the PRS, x^0 after the PRS, and the initial current residual z^0, which is calculated as $y - S(x^0)$. A PLV-Block consists of TSM and PPM which correspond to the thresholding operation and projection operation in the PL algorithm, respectively. In the kth PLV-Block, x^{k-1} is input into the TSM to capture richer image details and eliminate image noise. In addition, $\hat{x}^k$ after the TSM, y, and z^{k-1} are input into the PPM, where y is used to calculate the measurement residuals, and z^{k-1} is utilized for algorithm iteration. Subsequently, the updated x^k and z^k serve as the inputs for the next PLV-Block. With the increase of PLV-Blocks, the reconstruction process is progressively enhanced, leading to high-quality reconstruction. Finally, the image residuals $[r^1, \ldots, r^K]$ produced by all the PLV-Blocks and the output x^K of the last PLV-Block are input into the RIM to generate the final reconstructed image *Result*.

3.3. Preliminary reconstruction subnet

In traditional CS problems, the image sampling process can be expressed as $y = \boldsymbol{\Phi} x$. Similar to the sampling process in CSNet$^+$ [18], our sampling subnet considers each row of the sampling matrix as an individual convolutional filter to automatically update the sampling matrix. After the entire image is entered into the sampling subnet, a convolution layer is used to simulate the sampling process. This process can be formulated as:

$$y = S(x) \tag{6}$$

where $x \in \mathbb{R}^{H \times W \times 1}$ is the original image. During the sampling process, each original image is divided into $B \times B$ image blocks, which are then

sampled separately. When the CS ratio is $\frac{M}{N}$, non-overlapping sampling of the original image is achieved by using a convolutional layer $S(\cdot)$ with $B^2\frac{M}{N}$ convolutional kernels to replace the sampling matrix. The size of the convolutional kernel is $B \times B$, with a stride of B, resulting in $y \in \mathbb{R}^{\frac{H}{B}\times\frac{H}{B}\times B^2\frac{M}{N}}$ eventually. To preserve linearity, there are no bias terms or activation functions in the sampling network $S(\cdot)$.

The preliminary reconstruction process is illustrated by the PRS in Fig. 1, involving upsampling layer $U(\cdot)$ and image reshaping layer $R(\cdot)$. This process can be represented as:

$$x^0 = R(U(y)) \tag{7}$$

where $U(\cdot)$ is a convolutional layer with B^2 convolutional kernels, without adding bias terms and activation functions. The size of the convolutional kernel is 1×1, with a stride of 1. $U(\cdot)$ expands the channel size of the measurement y, transforming its shape into $U(y) \in \mathbb{R}^{\frac{H}{B}\times\frac{H}{B}\times B^2}$. Finally, the preliminary reconstructed image $x^0 \in \mathbb{R}^{H\times W\times 1}$ is produced through reshaping layer $R(\cdot)$, which sequentially reshapes B^2 channels into an image block of size $B \times B$.

3.4. Deep reconstruction subnet

Since the preliminary reconstruction image x^0 is usually of low quality, the DRS of the PLV-CSNet is used to improve the reconstruction by unfolding the PLV algorithm. The DRS primarily consists of multiple PLV-Blocks and an RIM, where each PLV-Block corresponds to one iteration of the PLV algorithm. Within the PLV-Block, the unfolded AMP algorithm is utilized to compute the projections of the approximation solution for the original image. Additionally, the DB is used to acquire and fuse rich multi-dimensional features, and then the soft thresholding method is used to perform denoising on the reconstructed image to enhance the reconstruction quality. The RIM is used to complement the image residuals reconstructed from measurement residuals generated during the unfolded algorithm and to eliminate artifacts and redundant information after the fusion of residuals, resulting in a smoother image.

It is worth noting that the preliminary reconstructed image achieved by the PRS is in insufficient resolution, and the corresponding noises would significantly impact the results of the following PPM. Therefore, within the PLV-Block, the order of the PLV algorithm is altered by denoising firstly with TSM and then obtaining an approximate solution via PPM. The following parts will provide a comprehensive introduction to the PLV-Block and the RIM.

3.4.1. Projected Landweber Variant Block (PLV-Block)

The PLV-Block consists of TSM and PPM which correspond to the thresholding operation and projection operation respectively, as illustrated in Fig. 1. Iteratively executing these two steps can progressively eliminate the unwanted noise from the reconstructed image and supplement the detailed texture information. The kth PLV-Block takes the output reconstructed image x^{k-1}, the current residual z^{k-1} from the previous PLV-Block and the measurement y obtained from PRS as its input, and then processes these inputs sequentially through the kth TSM and PPM to produce the reconstructed image x^k, the current residual z^k and the measurement residual r^k. The input of the first PLV-Block contains the preliminary reconstructed image x^0 and z^0.

3.4.1.1 Thresholding module (TSM): As shown in Fig. 2, the TSM consists of two stages. In the first stage, the output x^{k-1} of the $(k-1)$th PLV-Block are input to produce a feature map with 64 channels by F^{ext}. F^{ext} extracts features from images, generating high-dimensional features from local receptive fields. Then, the feature map is input to $DB(\cdot)$ to compress the feature map by denselayer and fuse the feature map by concatenation. In DB, the number of channels in the multi-dimensional feature map includes 64, 80, 96, 112, and 128. Additionally, the number of channels in the denselayer's feature map is 16. $DB(\cdot)$ integrates multi-dimensional image features including low-dimensional, mid-dimensional, and high-dimensional features. The

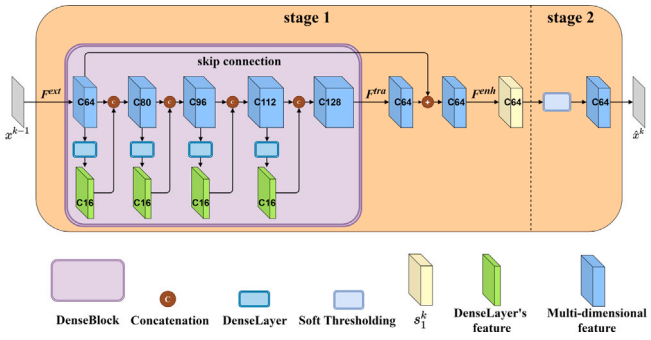


Fig. 2. Overview of the proposed TSM in kth PLV-Block.

feature map with 128 channels is reduced the number of channels in the feature maps to match the number of previous feature map with 64 channels by F^{tra}, followed by summation through skip connections. Finally, F^{enh} is utilized to enhance useful information in the input feature map, capturing complex patterns and features. Following this stage, feature map $s_1^k \in \mathbb{R}^{H\times W\times 64}$ is derived from x^{k-1} as input for the second stage. In the second stage, the noise of s_1^k is eliminated by soft thresholding. Then, a convolution layer is used to reduce the channel number of the feature map to 1 to output the reconstructed result $\hat{x}^k$.

The process of the first stage can be represented as:

$$s_1^k = F^{enh}(F^{tra}(DB(F^{ext}(x^{k-1}))) + F^{ext}(x^{k-1})) \tag{8}$$

where F^{enh}, F^{tra} and F^{ext} are constructed of one convolutional layer and one parametric rectified linear unit (PReLU), while F^{tra} additionally has a batch normalization. $DB(\cdot)$ consists of four denselayers and four concatenations for feature compression and feature fusion. Each denselayer is composed of two convolution layers, two batch normalizations, and two rectified linear units (ReLUs). The first convolution has a kernel size of 1×1 and the second convolution has a kernel size of 3×3, both with bias terms. After the denselayer compresses the dimension of feature maps, the denselayer's feature maps are obtained, thereby improving computational efficiency. The feature fusion operation concatenates the compressed denselayer's feature maps with the previous multi-dimensional feature maps. Following four rounds of feature compression and fusion, the channel number of the feature map is expanded to double the original feature map s_1^k, resulting in a shape $\mathbb{R}^{H\times W\times 128}$. The weights are shared in all PLV-Blocks to reduce parameters, accelerate training, reduce memory requirements, and enhance generalization capabilities.

In the second stage, the threshold parameter τ^k within the TSM is different and is independently updated. The process of the second stage can be represented as:

$$\hat{x}^k = Conv(soft(s_1^k, \tau^k)) \tag{9}$$

where the input s_1^k is initially denoised via soft thresholding algorithm, and then the channel number is compressed through a convolutional layer with a kernel size of 3×3 to produce a smoothed image $\hat{x}^k$.

3.4.1.2 Progressive projecting module (PPM): In the designed PPM, the equation for the kth iteration after combining neural network with the AMP algorithm is:

$$\begin{cases} z^k = y - S(\hat{x}^k) + \theta^k z^{k-1} \\ \check{x}^k = R(U(z^k)) \\ x^k = \hat{x}^k + \eta^k \check{x}^k \end{cases} \tag{10}$$

where η^k and θ^k represent two learnable scale factors. As shown in Fig. 3, The $S(\cdot)$ is used to sample the output $\hat{x}^k$ of TSM, then compute $S(\hat{x}^k)$ with $\theta^k z^{k-1}$ and y to obtain z^k, where θ^k represents the learnable scale factor of the Onsager reaction item in the kth PPM. Subsequently,

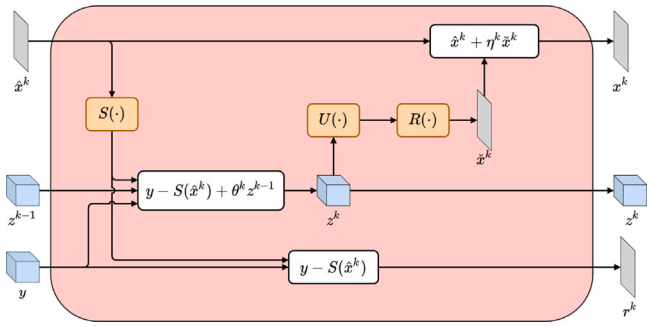


Fig. 3. Overview of the proposed PPM in kth PLV-Block.

z^k is inputted into $U(\cdot)$ and $R(\cdot)$ to obtain the $\check{x}^k$. Finally, the reconstructed image x^k of the kth PLV-Block is obtained by linearly shrinkage of the $\check{x}^k$ using another learnable scale factor η^k.

In order to obtain the measurement residuals, firstly, the reconstructed image in each iteration step of the deep reconstruction subnet is sampled, then the measurement after sampling are subtracted from the measurement in preliminary reconstruction subnet. Finally the measurement residuals are employed to reconstruct the image residuals. This approach fully leverages the additional information in the measurement, allowing them to play a more integral role in the image reconstruction process. For the kth iteration, the measurement residual output is:

$$r^k = y - S(\hat{x}^k) \tag{11}$$

where $\hat{x}^k$ represents the output of TSM. After the measurement residuals being calculated, they are then input into the RIM module for reconstructing the measurement residuals to the image residuals. The PPM repeatedly utilizes the measurement to calculate the measurement residuals, thereby extracting more comprehensive information from the measurement.

3.4.2. Residual Integration Module (RIM)

In all PLV-Blocks, operations such as feature extraction, feature fusion, and feature compression are applied to the image, which leads to the loss of certain image information and details during reconstruction. Hence, the measurement residual generated by each PLV-Block should be reconstructed to image residuals and then integrated into the reconstructed image to get improved reconstruction quality. However, there may be some artifacts in the reconstructed image. Additionally, supplementing multiple image residuals could lead to redundant image information.

To address these issues, the RIM is designed, which not only supplements the lost information in the reconstructed image but also eliminates artifacts by $D(\cdot)$ subnet and redundant information by $P(\cdot)$ subnet, resulting in a smoother reconstructed image. In RIM, the measurement residuals $[r^1, \ldots, r^K]$ output by PLV-Blocks are reconstructed into image residuals $[\hat{r}^1, \ldots, \hat{r}^K]$ through $U(\cdot)$ and $R(\cdot)$ first. The reconstruction process of kth image residual is as follows:

$$\hat{r}^k = R(U(r^k)) \tag{12}$$

As the measurement residuals generated by different PLV-Blocks vary, causing differences among image residuals, our RIM uses different regularization parameters $[\sigma^1, \ldots, \sigma^K]$ to weight each image residuals $[\hat{r}^1, \ldots, \hat{r}^K]$. These regularization parameters are continuously updated through the neural network's backpropagation, allowing flexible integration and supplementation of various image residuals into the reconstructed image. The process can be formulated as:

$$x_f = D(x^K) + \sigma^1 \hat{r}^1 + \cdots + \sigma^K \hat{r}^K \tag{13}$$

$$Result = x_f - P(x_f) \tag{14}$$

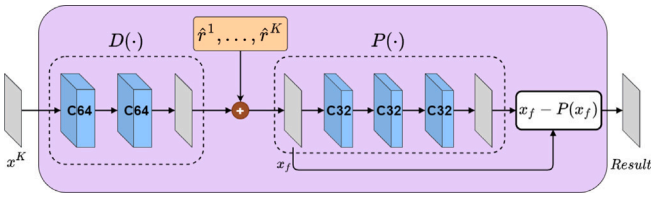


Fig. 4. Overview of the proposed RIM. $D(\cdot)$ and $P(\cdot)$ represent artifact elimination and redundancy elimination modules, respectively. The number of feature map channels in $D(\cdot)$ and $P(\cdot)$ is 64 and 32.

where $D(\cdot)$ is for artifact elimination module, which consists of three convolutional layers and two PReLU activation layers. The first convolutional layer has a kernel size of 7×7, while others have a size of 3×3. $P(\cdot)$ is for redundancy elimination module, which comprises four convolutional layers and three PReLU activation layers, with each convolutional layer having a kernel size of 3×3. To ensure the output has the same size as the input, padding sizes are set to 3 and 1 within $D(\cdot)$ and $P(\cdot)$, respectively. The structure of RIM is illustrated in Fig. 4. First, the reconstructed image x^K from the final PLV-Block is inputted into the $D(\cdot)$. Subsequently, the K adaptive regularization parameters are weighted into the K image residuals, and the weighted K image residuals $[\hat{r}^1, \ldots, \hat{r}^K]$ are added to $D(x^K)$ to obtain the integrated image x_f according to Eq. (13). Finally, the output of $P(\cdot)$ is subtracted from x_f to remove redundant information, further enhancing the reconstruction quality. In summary, the DRS of PLV-CSNet is outlined as Algorithm 1.

Algorithm 1 Deep Reconstruction Subnet of PLV-CSNet

Input: measurement y, preliminary reconstructed image x^0, the number of iteration K, threshold parameters $\tau^{1 \backsim K}$, scale factors $\theta^{1 \backsim K}$ and $\eta^{1 \backsim K}$, weight parameters $\sigma^{1 \backsim K}$
Output: reconstructed image $Result$

Initialization: $z^0 = y - S(x^0)$
Parameters: $\tau^{1 \backsim K}$, $\theta^{1 \backsim K}$, $\eta^{1 \backsim K}$
while $k \leq K$ **do**
  $s_1^k = F^{enh}(F^{tra}(DB(F^{ext}(x^{k-1}))) + F^{ext}(x^{k-1}))$
  $\hat{x}^k = Conv(soft(s_1^k, \tau^k))$
  $z^k = y - S(\hat{x}^k) + \theta^k z^{k-1}$
  $\check{x}^k = R(U(z^k))$
  $x^k = \hat{x}^k + \eta^k \check{x}^k$
  $\hat{r}^k = R(U(r^k))$
  $r^k = y - S(\hat{x}^k)$
  $k \leftarrow k + 1$
end while
$x_f = D(x^K) + \sigma^1 \hat{r}^1 + \ldots + \sigma^K \hat{r}^K$
$Result = x_f - P(x_f)$

3.5. Loss function

In the training phase, the mean square error is adopted as the loss function of PLV-CSNet. Given the training dataset $\{x_i\}_{i=1}^{N_b}$, N_b represents the total number of original images in the training set. We first use the real images $\{x_i\}_{i=1}^{N_b}$ as the inputs to PRS to generate preliminary reconstruction results, and then input these preliminary reconstruction results into DRS to obtain more precise and high-quality reconstruction results. The loss function can be expressed as:

$$L(\omega_1, \omega_2) = \frac{1}{N_b} \sum_{i=1}^{N_b} \| DRS(PRS(x_i, \omega_1); \omega_2) - x_i \|_2^2 \tag{15}$$

where ω_1 and ω_2 are network parameters for training in PLV-CSNet, corresponding to the learnable parameters in PRS and DRS. Minimizing the loss function using adaptive moment estimation (Adam) [53] results in the optimal values for these parameters.

4. Experiments

In this section, we introduce datasets and training details in Section 4.1, performance comparisons in Section 4.2 and ablation Studies in Section 4.3.

4.1. Datasets and training details

Following Ref. [20], a total of 400 images with 200 training images and 200 testing images from the BSDS500 dataset [54] are utilized as the training dataset. Before training, all training images are converted to grayscale and randomly cropped to the size of 96 × 96, resulting in 89 600 original images for network training. Moreover, augmentation techniques such as rotation and flipping are employed to further diversify the quantity and types of training data. Within the given CS ratios 1%, 4%, 10%, 25%, and 50%, PLV-CSNet is individually trained to conduct adaptive sampling and reconstruction on images. Our method is evaluated on widely used benchmark datasets, including Set5 [55], Set11 [17], BSD68 [56], and Urban100 [57], which have 5, 11, 68, and 100 images respectively. During testing, whole images without cropping are inputted into the network for end-to-end reconstruction. Peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) [58] are used as quantitative evaluation metrics for experimental results.

The calculation process of PSNR is:

$$\mathrm{MSE} = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W}[Result(i,j) - x(i,j)]^2 \tag{16}$$

$$\mathrm{PSNR} = 10 \cdot \log_{10}\left(\frac{255^2}{\mathrm{MSE}}\right) \tag{17}$$

where MSE is the mean squared error between the reconstructed image and original image.

The calculation process of SSIM is:

$$\mathrm{SSIM} = \frac{(2\varepsilon_1\varepsilon_2 + C_1)(2\varphi_{12} + C_2)}{(\varepsilon_1^2 + \varepsilon_2^2 + C_1)(\varphi_1^2 + \varphi_2^2 + C_2)} \tag{18}$$

ε_1 and ε_2 represent the average of the original image and reconstructed image, respectively. φ_1 and φ_2 represent the variance of the original image and reconstructed image, respectively. φ_{12} is the covariance of the reconstructed image and original image. C_1 and C_2 are two constants to control division.

In PLV-CSNet, the number of PLV-Blocks is set to $K = 9$, hence the number of PPM and TSM are also 9. The threshold parameters $\tau^{1\sim K}$ for soft thresholding in the kth PLV-Block are initialized to 0.01. The scale factor $\theta^{1\sim K}$ and $\eta^{1\sim K}$ for the projecting module are both initialized to 0.5. The weight parameters $\sigma^{1\sim K}$ in RIM are all initialized to 0.5. To ensure smooth convergence during optimization, the batch size for training is set to 16. The learning rate of the first 40 epochs is 0.0002 and is reduced to half in each following 40 epochs, respectively. Regarding the network implementation, the entire PLV-CSNet is written and debugged based on the Python platform's pytorch. In experiments, it takes approximately five days to complete one full network model training on CPU Intel Core i7-11700 and GPU GTX2080ti. The entire training process spans 300 epochs. Table 1 summarizes the hyperparameters used by PLV-Network in the experiments.

Table 1
List of hyperparameters.

Hyperparameters	Values
Initial $\tau^{1\sim K}$	0.01
Initial $\theta^{1\sim K}$	0.5
Initial $\eta^{1\sim K}$	0.5
Initial $\sigma^{1\sim K}$	0.5
Batch size	16
Learning rate	0.0002
Training epoch	300
Number of PLV-Block	9
Number of training image	89 600
Input image size	96 × 96

4.2. Performance comparisons

4.2.1. Objective comparisons

The proposed PLV-CSNet is compared with several representative CS reconstruction networks, including ISTA-Net$^+$ [35], CSNet$^+$ [18], iPiano-Net [59], COAST [39], OPINE-Net$^+$ [38], AMP-Net [40], TransCS [41], TCS-Net [34] and OCTUF [43]. Among them, CSNet$^+$ and TCS-Net are pure data-driven networks, while ISTA-Net$^+$, iPiano-Net, OPINE-Net$^+$, COAST, AMP-Net, TransCS and OCTUF are deep unfolding networks. The codes for these comparison algorithms are downloaded from the authors' websites.

In order to verify the reconstruction quality of PLV-CSNet under different CS ratios, extensive experiments are conducted on five CS ratios 1%, 4%, 10%, 25%, and 50%. Table 2 presents the average PSNR/SSIM results of various methods on Set5, Set11, BSD68, and Urban100 datasets, respectively, and the best results are highlighted in bold. It can be clearly seen that, at low CS ratios, the proposed PLV-CSNet outperforms all competing networks in terms of PSNR/SSIM. Compared to the pure data-driven networks, the proposed PLV-CSNet exhibits certain PSNR improvements across all CS ratios for all datasets. Furthermore, the proposed PLV-CSNet demonstrates superior reconstruction quality compared to other deep unfolding networks, including ISTA-Net$^+$, iPiano-Net, COAST, OPINE-Net$^+$, AMP-Net, and TransCS. Since recent methods such as COAST, OPINE-Net$^+$, and TransCS can provide good reconstruction performance compared with other existing methods, the experimental results of PLV-CSNet are compared with these three CS methods in various test datasets. In the Set5 test dataset, our proposed PLV-CSNet could achieve improvements of 1.50, 2.97, 2.05 in PSNR and 0.0210, 0.0755, 0.0974 in SSIM at CS ratio of 1%. In the Set11 test dataset, the proposed PLV-CSNet could achieve improvements of 0.92, 0.85, 1.07 in PSNR and 0.0175, 0.0231, 0.0243 in SSIM at a CS ratio of 4%. In the BSD68 test dataset, the PLV-CSNet could achieve improvements of 0.66, 0.68, 0.33 in PSNR and 0.0174, 0.0226, 0.0381 in SSIM at a CS ratio of 10%. In the Urban100 test dataset, the PLV-CSNet achieved improvements of 0.52, 0.70, 0.55 in PSNR and 0.0271, 0.0096, 0.0055 in SSIM at CS ratio of 25%. In Table 2, it can be observed that the deep unfolding network OCTUF outperforms the proposed PLV-CSNet when CS ratios are 25% and 50% in Set11, BSD68, and Urban100 datasets. While the CS ratios decrease, the proposed PLV-CSNet achieves much better reconstruction performance against OCTUF. The possible explanation is provided as follows: When CS ratios are 1% and 4%, the guidance is greatly limited because the dimension of the sampling matrix is very low. In this cases, limited information makes it difficult to achieve high-quality image reconstruction. However, our network repeatedly utilizes the measurement to compute the image residuals, extracting more comprehensive information, which enhances the network's ability to reconstruct images. As a result, at CS ratios of 1% and 4%, PLV-CSNet outperforms OCTUF in reconstruction quality. OCTUF learns and integrates image features from different levels in parallel by outputting multiple channel images at each reconstruction stage. Different channel focuses on different aspects of the image, allowing the network to fully leverage the rich information. However, PLV-CSNet outputs only a single channel image at each reconstruction stage, which may lead to information loss or insufficient integration when processing highly compressed data. Therefore, at a CS ratio higher than 10%, the reconstruction performance of OCTUF is better than that of PLV-CSNet.

4.2.2. Subjective comparisons

To demonstrate the performance of the proposed method, our method is visually compared with several representative image CS reconstruction methods at different CS ratios. The reconstruction results of OPINE-Net$^+$ [38], AMP-Net [40], TransCS [41], TCS-Net [34],

Table 2
The PSNR/SSIM performance comparisons among various CS methods.

Dataset	Methods	CS ratio				
		1%	4%	10%	25%	50%
Set5	ISTA-Net+ [35]	18.55/0.4408	23.45/0.6240	28.63/0.8325	34.17/0.9277	39.49/0.9706
	CSNet+ [18]	24.18/0.6478	28.22/0.8091	32.22/0.9112	36.35/0.9577	41.61/0.9832
	iPiano-Net [59]	19.16/0.4503	23.72/0.6973	28.49/0.8534	33.87/0.9548	38.97/0.9788
	COAST [39]	23.31/0.6514	29.13/0.8499	33.90/0.9266	38.21/0.9648	43.15/0.9824
	OPINE-Net+ [38]	21.84/0.5969	27.84/0.8268	32.39/0.9147	36.73/0.9559	41.54/0.9802
	AMP-Net [40]	22.67/0.6158	27.97/0.8134	32.13/0.8996	36.71/0.9514	41.65/0.9790
	TransCS [41]	22.76/0.5750	29.02/0.8420	33.56/0.9244	38.26/0.9652	43.42/0.9850
	TCS-Net [34]	22.75/0.6001	27.55/0.8169	31.48/0.9063	35.85/0.9556	38.85/0.9710
	OCTUF [43]	23.01/0.6399	28.79/0.8459	33.22/0.9246	37.79/0.9635	42.85/0.9846
	PLV-CSNet	**24.81/0.6724**	**29.70/0.8591**	**34.15/0.9389**	**38.61/0.9673**	**43.22/0.9850**
Set11	ISTA-Net+ [35]	17.45/0.4131	21.56/0.6240	26.59/0.8073	32.39/0.9236	38.07/0.9706
	CSNet+ [18]	20.67/0.5411	24.83/0.7480	28.90/0.8840	33.51/0.9455	39.26/0.9807
	iPiano-Net [59]	20.16/0.5404	24.72/0.7696	29.49/0.8812	34.56/0.9568	39.97/0.9802
	COAST [39]	20.74/0.5681	25.56/0.7951	29.69/0.8818	34.98/0.9507	39.94/0.9744
	OPINE-Net+ [38]	20.24/0.5359	25.63/0.7895	29.76/0.8896	34.84/0.9516	40.17/0.9798
	AMP-Net [40]	20.36/0.5605	25.40/0.7735	29.35/0.8780	34.75/0.9487	40.26/0.9786
	TransCS [41]	20.32/0.5503	25.41/0.7883	29.54/0.8877	35.06/0.9548	40.49/0.9815
	TCS-Net [34]	21.09/0.5504	25.10/0.7406	28.64/0.8618	33.45/0.9436	36.90/0.9702
	OCTUF [43]	21.77/0.5936	26.43/0.8118	**30.72/0.9043**	**36.10/0.9601**	**41.34/0.9838**
	PLV-CSNet	**21.95/0.5988**	**26.48/0.8126**	30.49/0.9061	35.51/0.9577	40.91/0.9827
BSD68	ISTA-Net+ [35]	21.53/0.5098	25.03/0.6672	27.49/0.8090	31.11/0.9123	36.08/0.9698
	CSNet+ [18]	19.18/0.4201	22.34/0.5573	25.34/0.7031	29.28/0.8510	34.01/0.9421
	iPiano-Net [59]	20.67/0.4755	23.45/0.5913	26.33/0.7034	30.19/0.8532	34.88/0.9435
	COAST [39]	22.30/0.5391	25.34/0.6755	27.80/0.8091	31.81/0.9128	35.74/0.9597
	OPINE-Net+ [38]	21.93/0.5159	25.13/0.6824	27.78/0.8039	31.44/0.9047	36.25/0.9652
	AMP-Net [40]	21.23/0.5032	24.26/0.6559	26.85/0.7428	30.77/0.8543	35.87/0.9182
	TransCS [41]	22.05/0.5130	25.51/0.6683	28.13/0.7884	31.83/0.9011	36.78/0.9632
	TCS-Net [34]	22.26/0.5194	25.07/0.6829	27.42/0.8037	30.94/0.9082	35.67/0.9679
	OCTUF [43]	22.60/0.5356	25.56/0.6984	28.28/0.8183	**32.25/0.9187**	**37.41/0.9729**
	PLV-CSNet	**22.91/0.5465**	**25.67/0.7037**	**28.46/0.8265**	32.07/0.9172	36.98/0.9713
Urban100	ISTA-Net+ [35]	18.82/0.4200	21.65/0.6407	24.67/0.8080	29.08/0.9202	34.57/0.9728
	CSNet+ [18]	16.19/0.3168	22.34/0.5573	22.57/0.6982	28.15/0.8868	34.58/0.9661
	iPiano-Net [59]	18.35/0.4145	22.06/0.6551	25.67/0.7963	30.87/0.9157	36.22/0.9675
	COAST [39]	18.77/0.4365	22.29/0.6815	25.94/0.8035	31.10/0.9168	35.74/0.9597
	OPINE-Net+ [38]	18.60/0.4167	22.08/0.6658	25.71/0.8291	30.92/0.9343	36.91/0.9796
	AMP-Net [40]	18.91/0.4344	22.00/0.6426	25.34/0.8051	30.51/0.9259	36.51/0.9768
	TransCS [41]	18.66/0.4286	22.21/0.6783	25.78/0.8332	31.07/0.9384	36.68/0.9645
	TCS-Net [34]	18.84/0.4259	22.06/0.6673	25.09/0.8215	29.57/0.9285	33.43/0.9617
	OCTUF [43]	19.02/0.4434	22.65/0.7019	**26.86/0.8597**	**32.77/0.9527**	**39.05/0.9859**
	PLV-CSNet	**19.21/0.4512**	**22.71/0.7073**	26.08/0.8428	31.62/0.9439	36.86/0.9753

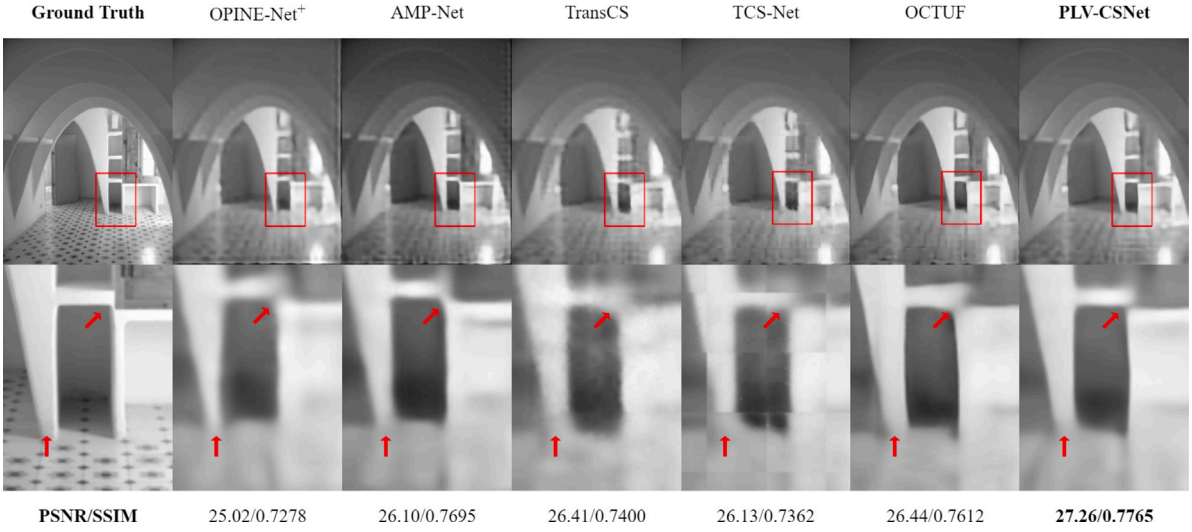


Fig. 5. Visual comparisons in terms of PSNR/SSIM on ROOM from Urban100 with CS ratio 1%.

OCTUF [43] and PLV-CSNet are displayed at CS ratios of 1% for Urban100, 4% for Set5, and 25% for Set11, respectively. Under image reconstruction with the low CS ratios of 1% and 4%, the results are illustrated in Figs. 5 and 6, where the optimal results are highlighted in bold. The red arrows are used to indicate the key comparison regions, highlighting the visual differences. It clearly shows that the

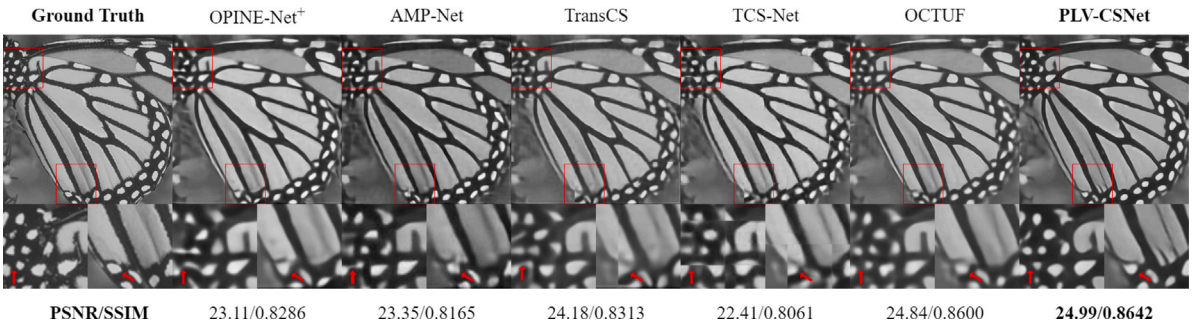


Fig. 6. Visual comparisons in terms of PSNR/SSIM on BUTTERFLY from Set5 with CS ratio 4%.

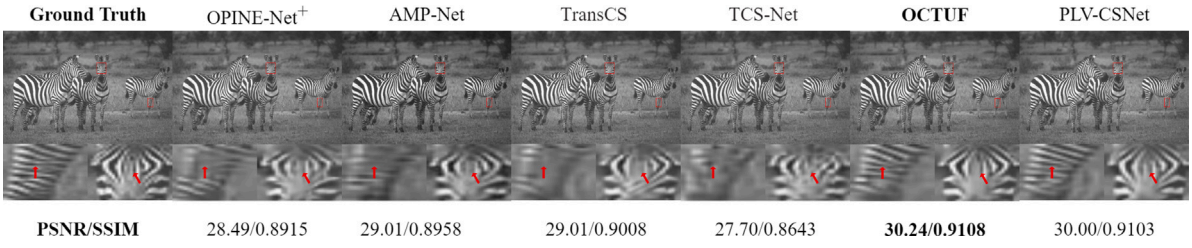


Fig. 7. Visual comparisons in terms of PSNR/SSIM on ZEBRA from BSD68 with CS ratio 25%.

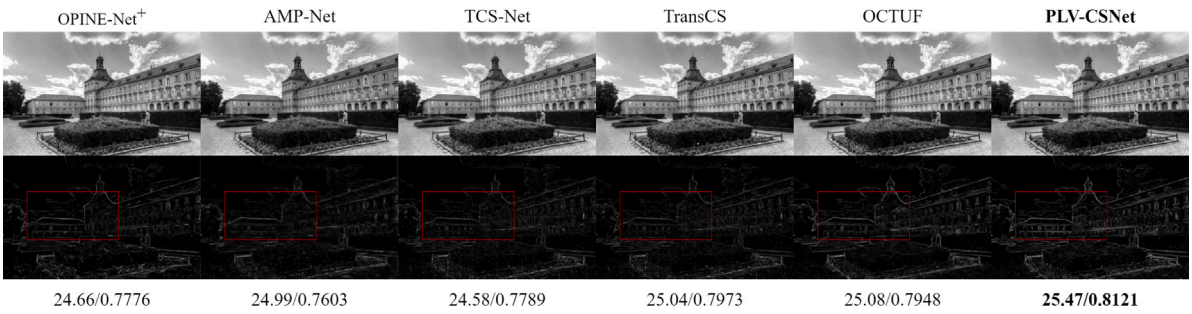


Fig. 8. Comparisons of various methods in recovering high-frequency details from the Urban100 dataset under CS ratio 10%.

reconstructed images by PLV-CSNet exhibit outstanding noise suppression capabilities. Furthermore, PLV-CSNet eliminates more artifacts and provides more reasonable reconstructions. Specifically, in the reconstruction of the ROOM image in Fig. 5, our network exhibits clearer contours and smoother edges. For TCS-Net, the patch-wise decoding phase focuses on reconstructing broader outlines, while the pixel-wise decoding phase refines textures. This multi-stage approach might inadvertently introduce inconsistencies or sharp transitions between patches, leading to artifacts in the output. TransCS uses a Transformer-based framework that emphasizes long-distance dependencies, which would result in over-smoothing within patches and insufficient blending at boundaries, accentuating artifacts.

For the CS ratio of 25%, as illustrated in Fig. 7, the visual reconstruction of the zebra image in the BSD68 dataset is presented. The enlarged local regions reveal the visual recovery effects of various methods on the zebra stripe details in the reconstructed images. Similar to Figs. 5 and 6, the red arrows are used to indicate the key comparison regions. A comparison of these regions indicates that PLV-CSNet and OCTUF demonstrate superior detail preservation compared to other models, with the reconstruction results closely resembling the original image. In the PLV-CSNet reconstructed image, noise suppression is notably more effective, resulting in higher clarity of the zebra stripes and smoother, sharper edge transitions. These advantages suggest that PLV-CSNet exhibits a stronger capability in maintaining detail fidelity, effectively reducing detail loss and blurring artifacts during the reconstruction process.

Table 3
Complexity comparison with competing methods.

Method	CS-Net+	OPINE-Net+	TransCS	TCS-Net	OCTUF	PLV-CSNet
#Parameters(M)	0.50	0.55	1.44	0.39	0.34	**0.21**
Size(MB)	1.92	2.12	20.4	1.55	4.11	**0.88**
Mem.(MB)	18.13	9.19	**5.39**	80.40	15.79	53.27

Moreover, several image CS reconstruction methods are evaluated on the Urban100 dataset to assess their capability in recovering high-frequency detail information. To facilitate a clear visual comparison, we extract the high-frequency components from the reconstructed images, thereby emphasizing the differences in detail. As illustrated in Fig. 8, the proposed PLV-CSNet consistently surpasses other competing methods. PLV-CSNet shows clearer details and sharper edges compared to those images recovered by other reconstruction methods. Especially in areas with complex lines, our proposed method demonstrates significant advances in reconstruction performance.

4.2.3. Complexity comparisons

The complexity of the proposed method is compared with the competing methods including CS-Net+ [18], OPINE-Net+ [38], TransCS [41], TCS-Net [34], and OCTUF [43]. Table 3 provides the comparisons of the parameters, the model size, and GPU memory for reconstructing a 96×96 image when CS ratio is 4%. Although our model requires some run-time memory, it has fewer parameters and a

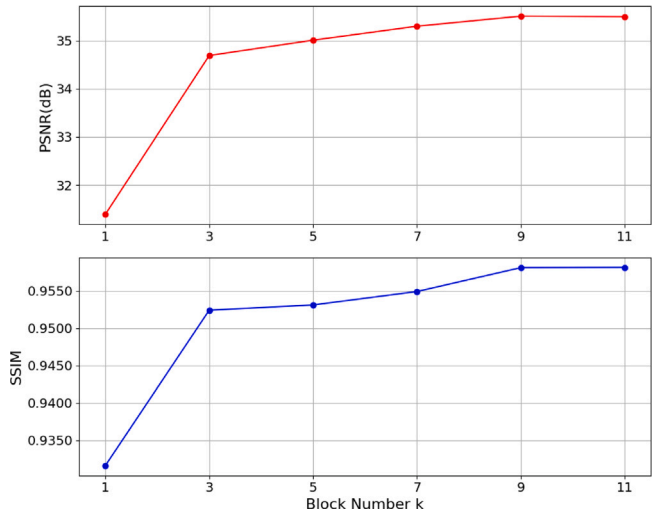


Fig. 9. Average PSNR and SSIM curves on Set11 achieved by PLV-CSNet versus PLV-Block numbers at the CS ratio 25%.

smaller model size compared to other CS reconstruction networks. This is mainly because DB typically use numerous feature map concatenations to preserve features and enhance performance, which increases storage requirements. However, by employing the weight sharing for DB within each TSM, we effectively reduce the number of parameters in the model, allowing the model to retain performance while achieving higher storage efficiency. From the results presented in Table 3, it is evident that our proposed PLV-CSNet demonstrates a significant reductions of 0.18M in model parameters and 0.67 MB in model size compared to the TCS-Net.

4.3. Ablation studies

In the ablation study, the impact of the number of PLV-Blocks is first explored, keeping other hyperparameters consistent with the previous network settings. Fig. 9 illustrates the PSNR and SSIM of PLV-CSNet with varying numbers of PLV-Blocks on Set11 dataset. One can observe that the PSNR and SSIM curves increase as the block number increases, and the curves become almost flat when block number $K \geq 9$. Balancing complexity and performance, we set the number of PLV-Blocks to 9 for our network.

To verify the effectiveness of TSM, PPM, and RIM, an ablation study is conducted on different module combinations of PLV-CSNet using the Set5 dataset at the CS ratio 25%. From Table 4, it can be observed that these three modules significantly impact the final performance of

Table 4
The reconstruction performance of different modules.

Module	Different combinations of modules in PLV-CSNet			
TSM	✗	✓	✓	✓
PPM	✓	✗	✓	✓
RIM	✓	✓	✗	✓
PSNR/SSIM	36.71/0.9580	37.51/0.9634	37.70/0.9629	**38.61/0.9673**

PLV-CSNet. The absence of TSM leads to a decrease of 1.90 dB and 0.0093 in the average PSNR and SSIM of reconstructed images by PLV-CSNet. Similarly, the absence of PPM results in a decrease of 1.10 dB and 0.0039, while the absence of RIM leads to a decrease of 0.91 dB and 0.0044 in the average PSNR and SSIM of reconstructed images by PLV-CSNet. It can be clearly found that all three modules have obvious gains and can improve the quality of reconstructed images.

Additionally, we conduct a visual analysis of the local regions and high-frequency details in the ablation experiments of the TSM, PPM, and RIM modules. As shown in Fig. 10, it is evident that without TSM, the image's stripes and edge details become blurry, and the structural information of the image is incomplete, causing a noticeable decline in reconstruction quality. Without PPM, the high-frequency details of the image appear blurry, and the texture details are difficult to restore clearly, especially in the edge regions where fine textures become distorted. In the absence of RIM, the high-frequency details of the image exhibit significant noise interference and discontinuities, leading to visible artifacts and rough edges in the zoomed-in regions. The complete PLV-CSNet, through the collaboration of all three components, achieves the optimal image reconstruction performance.

Finally, the convergence and performance of model are compared using the projection operation, hard thresholding, the unfolding AMP algorithm, and soft thresholding. The traditional PL algorithm consists of the projection operation and the hard thresholding. PLV-CSNet uses the AMP algorithm and the soft thresholding instead. The model using the projection operation and hard thresholding is referred to as "PH-CSNet", the model using the projection operation and soft thresholding is referred to as "PS-CSNet". Fig. 11 shows the comparison results of the models PH-CSNet, PS-CSNet, and PLV-CSNet on the Set5 test dataset with CS ratio 10%. This indicates that combining the AMP algorithm with the neural network to replace the projection operation in the PL algorithm can improve convergence speed and enhance reconstruction performance. Moreover, replacing hard thresholding with soft thresholding also contributes to improving reconstruction results.

5. Conclusion

Combining deep learning with PL Variant, this paper proposes a novel deep unfolding network PLV-CSNet for image CS reconstruction,

Fig. 10. Visual analysis of local regions and high-frequency details in PPM, TSM, and RIM ablation experiments.

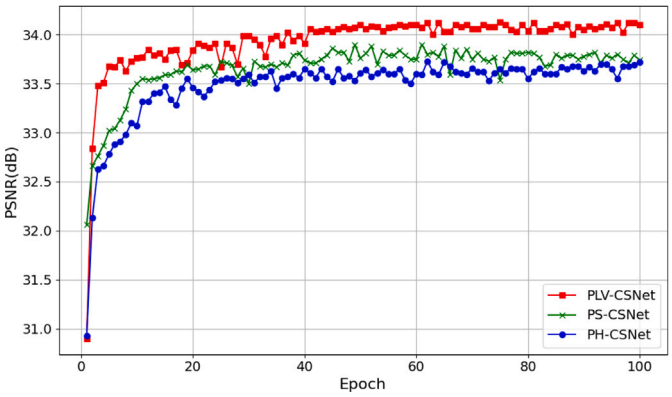


Fig. 11. The convergence and performance of the PLV-CSNet, PS-CSNet, and PH-CSNet Models (CS ratio 10%).

which has good mathematical properties and interpretability while further enhancing the reconstruction performance. The PLV-CSNet fully exploits the image multi-dimensional features obtained by the neural network in feature extraction, progressively achieves the approximate solution of the original signal, and addresses measurement residuals issues during algorithm iterations. The PLV-CSNet designs a PLV-Block consisting of TSM and PPM, together with the RIM. The PLV-Block is utilized as an iterative step of the PLV algorithm to progressively reconstruct images. In the PLV-Block, the TSM extracts image multi-dimensional features using dense blocks and eliminates image noise using soft thresholding. The PPM combines the AMP algorithm with the neural network to replace the projection operation and calculates the measurement residual generated at each iteration. The RIM reconstructs the measurement residuals into the image residuals and flexibly supplements them back to the reconstructed image. Experiments demonstrate that the proposed PLV-CSNet significantly enhances image quality and reduces network parameters compared to other image CS neural networks, and our method could perform well even at extremely low CS ratios. Additionally, ablation experiments are conducted to validate the effectiveness of each module in the network.

The PLV-CSNet demonstrates significant improvements in image reconstruction quality and reduces network parameters, and there are numerous works worthy of future exploration. First, within the DRS, the output of each reconstruction stage is currently a single channel image, which limits the model's ability to fully leverage information. Future improvements, inspired by OCTUF, could involve exploring multiple channel image for each stage, which would allow the network to process richer information and enhance reconstruction quality. Second, in the RIM, we aim to integrate more sophisticated techniques, such as attention mechanisms, to better capture and enhance the residuals, therefore improving the model's sensitivity to important features.

CRediT authorship contribution statement

Junpeng Hao: Writing – original draft, Investigation, Conceptualization. **Huang Bai:** Writing – original draft, Validation, Funding acquisition, Formal analysis. **Xiumei Li:** Writing – review & editing, Supervision, Methodology, Funding acquisition. **Marko Panic:** Writing – review & editing, Investigation. **Junmei Sun:** Writing – review & editing.

Declaration of competing interest

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

Data availability

Data will be made available on request.

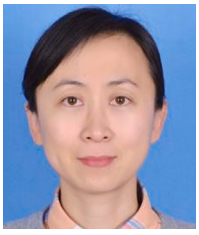
References

[1] D.L. Donoho, Compressed sensing, IEEE Trans. Inform. Theory 52 (4) (2006) 1289–1306.
[2] E.J. Candès, T. Tao, Near-optimal signal recovery from random projections: universal encoding strategies? IEEE Trans. Inf. Theory 52 (12) (2006) 5406–5425.
[3] G. Yang, S. Yu, H. Dong, G. Slabaugh, P.L. Dragotti, X. Ye, F. Liu, S. Arridge, J. Keegan, Y. Guo, D. Firmin, DAGAN: Deep de-aliasing generative adversarial networks for fast compressed sensing MRI reconstruction, IEEE Trans. Med. Imaging 37 (6) (2018) 1310–1321.
[4] Y. Li, W. Dai, J. Zou, H. Xiong, Y.F. Zheng, Structured sparse representation with union of data-driven linear and multilinear subspaces model for compressive video sampling, IEEE Trans. Signal Process. 65 (19) (2017) 5062–5077.
[5] Z. Wu, Z. Zhang, J. Song, M. Zhang, Spatial–temporal synergic prior driven unfolding network for snapshot compressive imaging, in: IEEE International Conference on Multimedia and Expo, ICME, 2021.
[6] X.J. Liu, S.T. Xia, F.W. Fu, Reconstruction guarantee analysis of basis pursuit for binary measurement matrices in compressed sensing, IEEE Trans. Inform. Theory 63 (5) (2017) 2922–2932.
[7] N. Nguyen, D. Needell, T. Woolf, Linear convergence of stochastic iterative greedy algorithms with sparse constraints, IEEE Trans. Inform. Theory 63 (11) (2017) 6869–6895.
[8] J.S. Lee, J.W. Choi, B. Shim, Sparse signal recovery via tree search matching pursuit, J. Commun. Netw. 18 (5) (2016) 699–712.
[9] S. Ji, Y. Xue, L. Carin, Bayesian compressive sensing, IEEE Trans. Signal Process. 56 (6) (2008) 2346–2356.
[10] Y. Liu, Y. Pang, X. Liu, Y. Liu, J. Nie, DIIK-Net: A full-resolution cross-domain deep interaction convolutional neural network for MR image reconstruction, Neurocomputing 517 (2023) 213–222.
[11] J. Dan, T. Jin, H. Chi, M. Liu, J. Yu, K. Cao, X. Yang, L. Zhao, H. Xie, PIRN: Phase invariant reconstruction network for infrared image super-resolution, Neurocomputing 599 (2024) 128221.
[12] L. Pan, G. Li, K. Xu, Y. Lv, W. Zhang, L. Li, L. Lei, Dual residual and large receptive field network for lightweight image super-resolution, Neurocomputing 600 (2024) 128158.
[13] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, F.E. Alsaadi, Deep-reinforcement-learning-based image segmentation for quantitative analysis of gold immunochromatographic strip, Neurocomputing 425 (2021) 173–180.
[14] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, F.E. Alsaadi, A survey of deep neural network architectures and their applications, Neurocomputing 234 (2017) 11–26.
[15] X. Chen, R. Yang, Y. Xue, C. Yang, B. Song, M. Zhong, A novel momentum prototypical neural network to cross-domain fault diagnosis for rotating machinery subject to cold-start, Neurocomputing 555 (2023) 126656.
[16] A. Mousavi, A.B. Patel, R.G. Baraniuk, A deep learning approach to structured signal recovery, in: 53rd Annual Allerton Conference on Communication, Control, and Computing, Allerton, Monticello, IL, USA, 2015, pp. 1336–1343.
[17] K. Kulkarni, S. Lohit, P. Turaga, R. Kerviche, A. Ashok, ReconNet: Non-iterative reconstruction of images from compressively sensed measurements, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Las Vegas, NV, USA, 2016, pp. 449–458.
[18] W. Shi, F. Jiang, S. Liu, D. Zhao, Image compressed sensing using convolutional neural network, IEEE Trans. Image Process. 29 (2020) 375–388.
[19] J. Chen, Y. Sun, Q. Liu, R. Huang, Learning memory augmented cascading network for compressed sensing of images, in: Lecture Notes in Computer Science, LNCS, 2020, pp. 513–529.
[20] D. Gunning, D.W. Aha, DARPA's explainable artificial intelligence program, Ai Mag. 40 (2) (2019) 44–58.
[21] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2016.
[22] M. Bertero, P. Boccacci, A. Koenig, Introduction to inverse problems in imaging, Opt. Photonics News 12 (10) (1998) 46–47.
[23] D. Sundman, S. Chatterjee, M. Skoglund, Greedy pursuits for compressed sensing of jointly sparse signals, in: European Signal Processing Conference, EUSIPCO, 2011, pp. 368–372.
[24] S.J. Kim, K. Koh, M. Lustig, S. Boyd, D. Gorinevsky, An interior-point method for large-scale l1-regularized least squares, IEEE J. Sel. Top. Signal Process. 1 (4) (2007) 606–617.
[25] I. Daubechies, M. Defrise, C. De Mol, An iterative thresholding algorithm for linear inverse problems with a sparsity constraint, Comm. Pure Appl. Math. 57 (11) (2004) 1413–1457.
[26] T. Erseghe, D. Zennaro, E. Dall' Anese, L. Vangelista, Fast consensus by the alternating direction multipliers method, IEEE Trans. Signal Process. 59 (11) (2011) 5523–5537.

[27] D.L. Donoho, A. Maleki, A. Montanari, Message-passing algorithms for compressed sensing, Proc. Natl. Acad. Sci. USA 106 (45) (2009) 18914–18919.
[28] C. Li, W. Yin, H. Jiang, Y. Zhang, An efficient augmented Lagrangian method with applications to total variation minimization, Comput. Optim. Appl. 56 (3) (2013) 507–530.
[29] C.A. Metzler, A. Maleki, R.G. Baraniuk, From denoising to compressed sensing, IEEE Trans. Inform. Theory 62 (9) (2016) 5117–5144.
[30] S. Mun, J.E. Fowler, Block compressed sensing of images using directional transforms, in: International Conference on Image Processing, ICIP, Cairo, Egypt, 2009, pp. 3021–3024.
[31] S. Mun, J.E. Fowler, Residual reconstruction for block-based compressed sensing of video, in: 2011 Data Compression Conference, Snowbird, UT, USA, 2011, pp. 183–192.
[32] J. Du, X. Xie, C. Wang, G.W. Shi, X. Xu, Y. Wang, Fully convolutional measurement network for compressive sensing image reconstruction, Neurocomputing 328 (2019) 105–112.
[33] H. Yao, F. Dai, S. Zhang, Y. Zhang, Q. Tian, C. Xu, DR^2-Net: deep residual reconstruction network for image compressive sensing, Neurocomputing 359 (2019) 483–493.
[34] H. Gan, M. Shen, Y. Hua, C. Ma, T. Zhang, From patch to pixel: A transformer-based hierarchical framework for compressive image sensing, IEEE Trans. Comput. Imaging 9 (2023) 133–146.
[35] J. Zhang, B. Ghanem, ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Salt Lake City, UT, USA, 2018, pp. 1828–1837.
[36] W. Dong, P. Wang, W. Yin, G. Shi, F. Wu, X. Lu, Denoising prior driven deep neural network for image restoration, IEEE Trans. Pattern Anal. Mach. Intell. 41 (10) (2019) 2305–2318.
[37] Y. Yang, J. Sun, H. Li, Z. Xu, ADMM-CSNet: A deep learning approach for image compressive sensing, IEEE Trans. Pattern Anal. Mach. Intell. 42 (3) (2020) 521–538.
[38] J. Zhang, C. Zhao, W. Gao, Optimization-inspired compact deep compressive sensing, IEEE J. Sel. Top. Signal Process. 14 (4) (2020) 765–774.
[39] D. You, J. Zhang, J. Xie, B. Chen, S. Ma, COAST: Controllable arbitrary-sampling network for compressive sensing, IEEE Trans. Image Process. 30 (2021) 6066–6080.
[40] Z. Zhang, Y. Liu, J. Liu, F. Wen, C. Zhu, AMP-Net: Denoising-based deep unfolding for compressive image sensing, IEEE Trans. Image Process. 30 (2021) 1487–1500.
[41] M. Shen, H. Gan, C. Ning, Y. Hua, T. Zhang, TransCS: A transformer-based hybrid architecture for image compressed sensing, IEEE Trans. Image Process. 31 (2022) 6991–7005.
[42] D. Ye, Z. Ni, H. Wang, J. Zhang, S. Wang, K. Sam, CSformer: Bridging convolution and transformer for compressive sensing, IEEE Trans. Image Process. 32 (2023) 2827–2842.
[43] J. Song, C. Mou, S. Wang, S. Ma, J. Zhang, Optimization-inspired cross-attention transformer for compressive sensing, in: IEEE AConference on Computer Vision and Pattern Recognition, CVPR, 2023, pp. 6174–6184.
[44] L. Xin, D. Wang, W. Shi, FISTA-CSNet: a deep compressed sensing network by unrolling iterative optimization algorithm, Vis. Comput. 39 (9) (2023) 4177–4193.
[45] J. Song, B. Chen, J. Zhang, Dynamic path-controllable deep unfolding network for compressive sensing, IEEE Trans. Image Process. 32 (2023) 2202–2214.
[46] W. Cui, X. Fan, J. Zhang, D. Zhao, Deep unfolding network for image compressed sensing by content-adaptive gradient updating and deformation-invariant non-local modeling, IEEE Trans. Multimed. 26 (2024) 4012–4027.
[47] J. Song, B. Chen, J. Zhang, Deep memory-augmented proximal unrolling network for compressive sensing, Int. J. Comput. Vis. 131 (6) (2023) 1477–1496.
[48] C. Zeng, Y. Yu, Z. Wang, S. Xia, H. Cui, X. Wan, GSISTA-Net: Generalized structure ISTA networks for image compressed sensing based on optimized unrolling algorithm, Multimedia Tools Appl. (2024) 1–15.
[49] J. Song, J. Zhang, SODAS-Net: Side-information-aided deep adaptive shrinkage network for compressive sensing, IEEE Trans. Instrum. Meas. 72 (2023) 1–12.
[50] X. Wang, H. Gan, UFC-Net: Unrolling fixed-point continuous network for deep compressive sensing, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, Seattle, WA, USA, 2024, pp. 25149–25159.
[51] G. Huang, Z. Liu, L. Van Der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Honolulu, HI, USA, 2017, pp. 2261–2269.
[52] S.G. Chang, B. Yu, M. Vetterli, Adaptive wavelet thresholding for image denoising and compression, IEEE Trans. Image Process. 9 (2000) 1532–1546.
[53] D. Kingma, J. Ba, Adam: A method for stochastic optimization, Comput. Sci. (2014).
[54] P. Arbeláez, M. Maire, C. Fowlkes, J. Malik, Contour detection and hierarchical image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 33 (5) (2011) 898–916.
[55] M. Bevilacqua, A. Roumy, C. Guillemot, M.-L.A. Morel, Low-complexity single-image super-resolution based on nonnegative neighbor embedding, in: Proceedings of the British Machine Vision Conference, 2012, pp. 1–10.
[56] D. Martin, C. Fowlkes, D. Tal, J. Malik, A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, in: Proceedings of Eighth IEEE International Conference on Computer Vision, ICCV, Vol. 2, Vancouver, BC, Canada, 2001, pp. 416–423.
[57] J.-B. Huang, A. Singh, N. Ahuja, Single image super-resolution from transformed self-exemplars, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Boston, MA, USA, 2015, pp. 5197–5206.
[58] A. Horé, D. Ziou, Image quality metrics: PSNR vs. SSIM, in: 2010 International Conference on Pattern Recognition, ICPR, Istanbul, Turkey, 2010, pp. 2366–2369.
[59] Y. Su, Q. Lian, iPiano-Net: Nonconvex optimization inspired multi-scale reconstruction network for compressed sensing, Signal Process., Image Commun. 89 (2020) 115989.

Junpeng Hao received his bachelor's degree in Computer Science and Technology from Zhejiang Gongshang University Hangzhou College of Commerce in 2022. He is currently pursuing his master degree in Computer Science and Technology from School of Information Science and Technology, Hangzhou Normal University. His research interests include deep learning and compressed sensing.

Huang Bai received the Ph.D. degree in control science and engineering from Zhejiang University of Technology, Hangzhou, China, in 2017. Since October 2017, he has been with School of Information Science and Technology, Hangzhou Normal University, Hangzhou, China. His research interests are in the areas of sparse and redundant representation, compressed sensing, and their applications to digital signal processing and wireless sensor network.



Xiumei Li received the Bachelor degree in electrical engineering from Lanzhou University, China, in 1999, the Master degree in information and signal processing from Institute of Acoustics, Chinese Academy of Sciences, China, in 2002, and the Ph.D. degree in information engineering from Nanyang Technological University, Singapore, in 2010. She has been with Hangzhou Normal University, China, since 2009, and currently is a professor in School of Information Science and Technology. Her research interests include compressed sensing and its applications, time frequency analysis and its applications, signal detection and estimation, and fast algorithms of signal processing methods.

Dr Marko Panić (M), is a research fellow at BioSense Institute, University of Novi Sad, Serbia. He received a double Ph.D. degree from the University of Novi Sad and Ghent University in 2020. His research interests include compressed sensing with multimodal imaging sensors with applications in biology, agriculture, environmental sciences, and health.

Junmei Sun received the Ph.D. degree from Shanghai University, Shanghai, China, in 2008. She is currently an Associate Professor and Master's Degree Supervisor with the School of Information Science and Technology, Hangzhou Normal University, Hangzhou, China. Her research interests include deep learning and intelligent software systems.