TDU-DLNet: A Transformer-Based Deep Unfolding Network for Dictionary Learning

Kai Wu[1], Jing Dong[1,*], Guifu Hu[1], Chang Liu[2], Wenwu Wang[3]

Abstract

Deep unfolding attempts to leverage the interpretability of traditional model-based algorithms and the learning ability of deep neural networks by unrolling model-based algorithms as neural networks. Following the framework of deep unfolding, some conventional dictionary learning algorithms have been expanded as networks. However, existing deep unfolding networks for dictionary learning are developed based on formulations with pre-defined priors, e.g., ℓ_1-norm, or learn priors using convolutional neural networks with limited receptive fields. To address these issues, we propose a transformer-based deep unfolding network for dictionary learning (TDU-DLNet). The network is developed by unrolling a general formulation of dictionary learning with an implicit prior of representation coefficients. The prior is learned by a transformer-based network where an inter-stage feature fusion module is introduced to decrease information loss among stages. The effectiveness and superiority of the proposed method are validated on image denoising. Experiments based on widely used datasets demonstrate that the proposed method achieves competitive results with fewer parameters as compared with deep learning and other deep unfolding methods.

Keywords: Deep unfolding network, dictionary learning, transformer networks, image denoising

[*] Corresponding author.

[1] K. Wu, J. Dong, and G. Hu are with the College of Electrical Engineering and Control Science, Nanjing Tech University, Nanjing, Jiangsu, China. (email: jingdong@njtech.edu.cn)

[2] C. Liu is with the College of Ocean Science and Engineering, Shandong University of Science and Technology, Qingdao, Shandong, China.

[3] W. Wang is with the Centre for Vision, Speech and Signal Processing, University of Surrey, Guildford, GU2 7XH, UK.

1. Introduction

Sparse representation provides effective prior information for inverse problems in signal and image processing by representing a signal $\mathbf{y} \in \mathbb{R}^m$ as linear combinations of a few atoms from a dictionary $\mathbf{D} \in \mathbb{R}^{m\times d}$, i.e., $\mathbf{y} = \mathbf{D}\mathbf{x}$, where the representation coefficient $\mathbf{x} \in \mathbb{R}^d$ is a sparse vector with many zero elements, and its sparsity is represented as the number of non-zero elements. Compared with pre-defined dictionaries, adaptive dictionaries learned from data tend to represent signals more accurately and achieve better performance. Many dictionary learning algorithms have been proposed [1, 2, 3, 4, 5] and found applications in image denoising, image super-resolution, signal declipping, and image deblurring [6, 7, 8, 9, 10]. However, traditional dictionary learning usually divides the whole image into over-lapping patches, which neglects the dependencies between the patches and lacks shift-invariance. To address this issue, convolutional dictionary learning (CDL) [11, 12, 13] replaces the matrix product operation with the convolution operation, which is shown to outperform the traditional dictionary learning in many applications, e.g., image denoising [14], image fusion [15], and rain streak removal [16]. It should be noted that both traditional dictionary learning and convolutional dictionary learning are model-based methods using explicit formulations for specific tasks.

In recent years, deep neural networks (DNN) have achieved better results than traditional model-based methods in many image processing tasks [17, 18, 19, 20, 21]. Driven by a large amount of data, DNN can learn deep structural information of images. However, DNN models usually lack good interpretability due to its "black-box" nature. In contrast, traditional model-based algorithms are usually highly interpretable, as they are developed via modeling the physical processes underlying specific problems. To integrate the advantages of DNN models and model-based algorithms, deep unfolding [22, 23, 24, 25, 26, 27, 28] has been proposed by developing model-driven DNNs. A widely used mechanism in deep unfolding is to unroll an iterative algorithm to an end-to-end network, where the optimal variables and parameters in the original model can be converted as parameters of a network and learned in a data-driven fashion via back-propagation [22, 25, 29, 30]. In addition, some researchers attempt to integrate more elements of neural networks to further improve the performance of deep unfolding methods. One approach is to employ convolutional layers as an alternative to matrix multiplications or convolutional computations in conventional algorithms [31, 32]. This can enhance the flexibility of the model and accelerate the convergence of the algorithm. Another approach plugs an existing neural network, e.g. UNet [33], into the unfolding framework [28, 34]. The plug-in of the neural network makes it possible to learn implicit priors from data, which can greatly improve the performance of the model.

However, existing dictionary learning methods based on deep unfolding still have some shortcomings for improvement. First, some of the methods adopt predefined priors with explicit definitions, e.g., ℓ_1-norm regularizer, in the models for unfolding [31, 32, 27]. Second, in the deep unfolding methods that learn priors from data, convolutional neural networks (CNNs) are often used [28], [34]. Nevertheless, CNNs have limited receptive fields due to the use of convolution operators, and thus may not be effective in modelling the long-range dependencies within data such as image patches. This may prevent the priors learned by CNN-based unfolding methods from capturing the global features from data [35]. Moreover, existing unfolding networks neglected the relationships between the features extracted in different stages of the network, and results in information loss in terms of feature extraction [34]. To solve these problems, we propose to unfold a general dictionary learning model with an implicit prior and develop a transformer-based deep unfolding framework for dictionary learning (TDU-DLNet). Our main contributions are summarized below:

1) The overall architecture of the proposed model is derived by unrolling a general model of dictionary learning using an implicit prior formed from the representation coefficients. The learning of the prior is realized by a transformer-based network which considers long-range dependencies be-

tween image patches.

2) An inter-stage feature fusion module is embedded to inherit the features learned in the previous stages and reduces the information loss between different stages of the unfolding model.

3) The proposed model is applied to image denoising and obtains better results than classic deep learning methods and other state-of-the-art deep unfolding methods.

The remaining sections are organized as follows. Section 2 presents some related work including the general model for dictionary learning and the deep unfolding methods. Section 3 introduces the formulation, alternating direction method of multipliers (ADMM) algorithm, and the corresponding deep unfolding architecture. Section 4 presents experimental details and results as compared with other models and Section 5 draws the main conclusions of the study.

2. Related Works

2.1. Dictionary learning

Dictionary learning aims to learn a dictionary from a set of training signals where the signals can be represented as linear combinations of a few atoms of the learned dictionary. Let $\mathbf{Y} \in \mathbb{R}^{m\times n}$ and $\mathbf{D} \in \mathbb{R}^{m\times d}$ denote the training signals and the dictionary to be learned, respectively. The general formulation of dictionary learning can be written as [1]

$$\arg\min_{\mathbf{D},\mathbf{X}} \frac{1}{2}\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2 + \lambda\varphi(\mathbf{X}), \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^{d\times n}$ denotes the representation coefficients of the signals $\mathbf{Y}$ with respect to the dictionary $\mathbf{D}$, and λ is the penalty parameter. The first term is the reconstruction error, where $\|\cdot\|_F$ denotes the Frobenius norm. The second term $\varphi(\mathbf{X})$ denotes the regularization function reflecting the sparse property of the representation coefficients. Most existing dictionary learning algorithms use ℓ_1 or ℓ_0 norm to constrain $\mathbf{X}$ [1, 36].

Dictionary learning was initially applied to image denoising by reconstructing images using a learned dictionary and the corresponding representation coefficients [37], and it has found applications in other inverse problems of image processing, such as image denoising, image super-resolution, image deblurring, and image segmentation [38, 7, 9, 10]. However, in these applications, the process of dictionary learning and image reconstruction is based on image patches, which results in dictionaries lacking the property of shift invariance. The dictionaries are sensitive to the translation of image features and cannot extract image patterns effectively [11, 12]. To address this issue, CDL [11, 12, 13] is proposed by substituting the dictionary with linear filters. In this way, an entire image can be represented directly using the convolution with the filters, and does not have to be broken down into small patches. This method yields outstanding performance in image reconstruction and denoising [39], [40]. Traditional dictionary learning and CDL are unsupervised tasks and their applications to image restoration are usually in an unsupervised manner without the access to clean images [37, 9, 11].

2.2. Deep unfolding

Deep unfolding constructs model-driven networks by unrolling traditional iterative model-based algorithms to end-to-end networks. In particular, the operations of each iteration are unfolded as one layer of a network, and the optimization variables and parameters are converted to the parameters to be learned in the network. The seminal work in [22] unrolls iterative algorithms to neural networks and develops neural network approximations for sparse coding. In [26], the traditional ADMM algorithm [41] is unrolled to a network for image compressive sensing. In [23], a traditional non-negative matrix factorization algorithm is unrolled to a deep network for single channel speech separation. Deep unfolding networks by unrolling image reconstruction models with sophisticated priors have been proposed. For example, edge-related priors are considered in [42, 43, 44] and priors in transform domains are exploited in the deep unfolding works [45, 46]. In [25], a comprehensive review for deep unfolding for signal and image processing is given, including popular unrolling techniques, algorithms, and their applications.

In the field of dictionary learning, Meyer et al. [27] reformulated the calculation chain of a classical dictionary learning algorithm K-SVD [1] and proposed an end-to-end framework, namely, the learned K-SVD (LKSVD). Bahareh et al. [47] develops a constrained neural network by unfolding the iterative optimization procedure for CDL. In [32], stride convolutional and transposed-convolutional layers are utilized to formulate the convolution operator in the traditional Convolutional Sparse Coding (CSC) model, which can also support the shift invariance as in CDL. Apart from the convolution layers, some other network structures are also employed to expand the iterative unfolding framework. For example, Zheng et al. [28] use the UNet [33] framework to learn the prior features of coefficient representation rather than using handcrafted priors, e.g., ℓ_1 norm, that are widely used in dictionary learning. Yan et al. [39] further improves this work and applies it for denosing low-dose CT images. By integrating the neural network framework into the iterative unfolding framework, many deep unfolding models [48, 49, 50] utilize this strategy to leverage the learning ability of neural networks. These prior works have demonstrated the effectiveness and the efficiency of the framework of deep unfolding [32, 47, 28], as compared with traditional iterative algorithms and deep learning. However, most of the current deep unfolding frameworks use convolutional neural networks as a plug-in, which is effective in capturing local features, but tends to be limited in capturing global information, e.g. the relations between pixels from distant receptive fields. Transformer has been used in deep unfolding networks [51], [52], [53], however, these networks are inspired by image restoration models rather than dictionary learning. In these methods, the unfolding architecture relies on formulations for image restoration, where the priors on image data are exploited to reconstruct the images directly.

Most deep unfolding networks are troubled with information loss between the stages due to the intrinsic feature-to-image transformation [34]. Some inter-stage feature fusion techniques have been developed to alleviate this issue. Mou et al. [34] and Dong et al. [52] proposed stage interaction modules in a spatially adaptive normalization manner [54]. Li et al. [51] performed stage feature fusion in the Fourier transform domain for reconstruction of hyperspectral images. However, these feature fusion modules are used in deep unfolding methods for image restoration. In the area of dictionary learning, no work has considered the fusion of features across different stages of deep unfolding networks.

3. Proposed Algorithm

In this section, we first consider a general model for dictionary learning and develop an iterative optimization method based on ADMM [41] to address the obtained problem. A network is then designed by unfolding the optimization model.

3.1. Problem Formulation and ADMM solver

We consider the general formulation (1) directly without defining any regularization terms explicitly. To decouple the optimization variables in the reconstruction term and the regularization term, we introduce an independent auxiliary variable $\mathbf{Z}$, and reformulate (1) as

$$\begin{aligned} \arg\min_{\mathbf{D},\mathbf{X},\mathbf{Z}} \ & \frac{1}{2}\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2 + \lambda\varphi(\mathbf{Z}) \\ \text{s.t.} \ & \mathbf{X}=\mathbf{Z}, \end{aligned} \tag{2}$$

where $\varphi(\mathbf{Z})$ represents the prior of $\mathbf{Z}$ without any explicit definitions. This formulation is different from the formulations used in the existing deep unfolding methods for dictionary learning. For example, LKSVD [27] and the method proposed in [47] utilize the ℓ_1-norm to constrain the representation coefficients rather than learn a prior from data. DCDicL [28] employs priors on both the dictionary and the representation coefficients. Compared with these works, formulation (2) applies only an implicit prior on the representation coefficients, and is more suitable to learn a general dictionary as the prior is learned from data.

As ADMM has strong theoretical guarantees even for non-convex and non-smooth problems [41, 55], we develop the unfolding network based on ADMM. The corresponding augmented Lagrangian function of the above problem is

$$\begin{aligned}\mathcal{L}_{\rho}(\mathbf{X},\mathbf{Z},\boldsymbol{\mu}) =& \frac{1}{2}\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2 + \lambda\varphi(\mathbf{Z}) + \langle\boldsymbol{\mu},\mathbf{X}-\mathbf{Z}\rangle \\ &+ \frac{\rho}{2}\left\|\mathbf{X}-\mathbf{Z}\right\|_F^2,\end{aligned} \tag{3}$$

where $\boldsymbol{\mu}$ is Lagrangian multipliers and ρ is a penalty parameter. To make the subsequent formulas more concise, we assume $\boldsymbol{\beta} = \frac{\boldsymbol{\mu}}{\rho}$. Based on the framework of ADMM [41], the original problem can be addressed by alternatively updating the variables $\mathbf{X}$, $\mathbf{D}$, $\mathbf{Z}$, and $\boldsymbol{\beta}$, i.e.,

$$\begin{cases} Solve_{\mathbf{X}} : \arg\min\limits_{\mathbf{X}} \frac{1}{2}\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2 + \frac{\rho}{2}\left\|\mathbf{X}+\boldsymbol{\beta}-\mathbf{Z}\right\|_F^2 \\ Solve_{\mathbf{D}} : \arg\min\limits_{\mathbf{D}} \frac{1}{2}\|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_F^2 \\ Solve_{\mathbf{Z}} : \arg\min\limits_{\mathbf{Z}} \frac{\rho}{2}\left\|\mathbf{X}+\boldsymbol{\beta}-\mathbf{Z}\right\|_F^2 + \lambda\varphi(\mathbf{Z}) \\ Solve_{\boldsymbol{\beta}} : \boldsymbol{\beta} \leftarrow \boldsymbol{\beta} + \eta\left(\mathbf{X}-\mathbf{Z}\right), \end{cases} \tag{4}$$

where η is the rate for updating the Lagrangian multiplier. Different from the projected gradient descent [56], ADMM uses an extra intermediate step to update the Lagrangian parameters in the iterative steps, i.e., $Solve_{\boldsymbol{\beta}}$, due to the employment of augmented Lagrangian for relaxation and variable splitting. This further improves the convergence rate of the algorithm.

We generalize the dictionary $\mathbf{D}$ as a convolutional layer $Conv_{\mathbf{D}}$ [31, 32, 34], where $\mathbf{D}$ is embedded as learning parameters in the convolutional layer, i.e., the convolution kernel. In this way, the learned dictionary can be applied to images of different sizes directly without resizing the images or extracting patches, and the update of $\mathbf{D}$ can be realized in the training of the unfolded network implicitly. The data $\mathbf{Y}$ can be an entire image, rather than a set of patches as in the traditional dictionary learning model. Similarly, the transpose of $\mathbf{D}$ is simulated using another convolutional layer $Conv_{\mathbf{D}^T}$, where the transpose of $\mathbf{D}$ acts as the convolution kernel. In fact, this simulation can be regarded as a generalization of the conventional dictionary. In conventional dictionary learning, signals can be represented using the dictionary and the representation coefficients, where the dictionary contains features that are learned from the signals. In the proposed model, the convolutional layers, which are used to simulate the dictionary and its transpose, are also used to extract the features.

The update of the other variables in the k-th iteration can be written as

$$\begin{aligned} Update_{\mathbf{X}} : \mathbf{X}^k = \mathbf{X}^{k-1} - \alpha^k \left\{ Conv^k_{\mathbf{D}^T} \left[Conv^k_{\mathbf{D}}(\mathbf{X}^{k-1}) - \mathbf{Y} \right] \right. \\ \left. + \rho^k \left(\mathbf{X}^{k-1} - \mathbf{Z}^{k-1} + \boldsymbol{\beta}^{k-1} \right) \right\}, \end{aligned} \tag{5a}$$

$$Update_{\mathbf{Z}} : \mathbf{Z}^k = \mathcal{S}\left[\mathbf{X}^k + \boldsymbol{\beta}^{k-1}, \lambda^k / \rho^k \right], \tag{5b}$$

$$Update_{\boldsymbol{\beta}} : \boldsymbol{\beta}^k = \eta^k_1 \boldsymbol{\beta}^{k-1} + \eta^k_2 \mathbf{X}^k - \eta^k_3 \mathbf{Z}^k, \tag{5c}$$

where $\mathcal{S}(.)$ denotes the proximal operator of $\varphi(\cdot)$ [26]. To improve the flexibility of the model, we generalize the single parameter η in (4) as three independent parameters η^k_n $(n = 1, 2, 3)$. In the conventional ADMM, the hyper-parameters α^k, β^k, λ^k, ρ^k, and η^k need to be pre-specified, while in the proposed deep unfolding network, these hyper-parameters are generalized as learnable parameters as will be detailed in the next section.

The update of $\mathbf{X}$ is based on a gradient descent step with α^k being the step size. As the dictionary $\mathbf{D}$ and its transpose are simulated using two convolutional layers, the gradient used in $Update_{\mathbf{X}}$ is also based on the convolutional layers, which can be seen as a prediction of the actual gradient [34]. In the update of $\mathbf{Z}$, the proximal operator $\mathcal{S}(.)$ is usually a soft thresholding function corresponding to the ℓ_1-norm regularizer which is widely used in traditional dictionary learning. In [26], the regularizer $\varphi(\cdot)$ is generalized as a learnable piecewise linear function rather than a fixed function. We attempt to learn a more general regularizer without explicit definitions by regarding the subproblem $Solve_{\mathbf{Z}}$ as a denoising problem, and use deep neural networks to address the problem.

By combining the ADMM iterations with the data-driven strategy and generalized denoiser, the iterative ADMM algorithm can be unfolded as a network. The details of the deep unfolding network will be presented in Section 3.2.

3.2. Deep unfolding architecture

The whole architecture of the proposed TDU-DLNet is shown in Fig. 1, which is constructed by unrolling the iterative steps of the ADMM solver in (5).

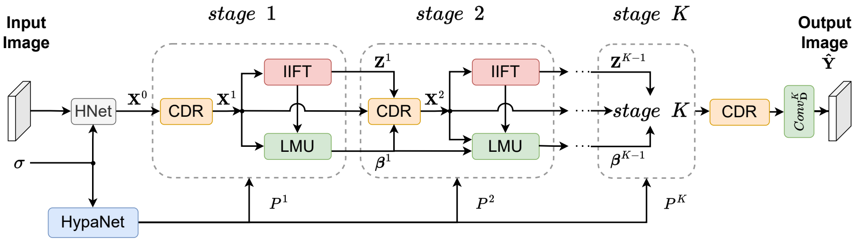


Figure 1: The whole deep unfolding architecture of TDU-DLNet.

The optimal variable and the regularization coefficient in (2) are transferred to the network parameters, and these parameters can be learned in an end-to-end fashion as in traditional neural networks. Each iteration of the iterative algorithm is seen as one stage of TDU-DLNet, and the main structure of TDU-DLNet consists of K repeated stages. Each stage has three interrelated modules: Convolutional Dictionary Reconstruction (CDR) module, Interstage Information Fusion Transformer (IIFT) module, and Lagrangian Multipliers Update (LMU) module. These three modules correspond to the update of $\mathbf{X}$, $\mathbf{Z}$, and $\boldsymbol{\beta}$ in equation (5), respectively. The update of the dictionary $\mathbf{D}$ is embedded in the convolutional layers of the CDR module, as mentioned in Section 3.1. As shown in Fig. 1, the CDR module converts the image to the dictionary domain by estimating the representation coefficients $\mathbf{X}$ so that the IIFT module can perform denoising in this new domain by estimating $\mathbf{Z}$ from $\mathbf{X}$. The LMU module connects various stages by fusing the output of CDR and IIFT in the current stage and passes this fused output to the next stage.

At the beginning of the proposed architecture, an HNet [28] is used to initialize the feature map $\mathbf{X}^0$ based on the noisy image and the noise level σ. HNet consits of 2 convolutional layers with the Rectified Linear Unit (ReLU) activation.

To improve the adaptability of the proposed model to different noise levels, we utilize HypaNet [28] to learn adaptive hyper-parameters with respect to noise levels. HypaNet is composed of 2 convolutional layers (kernel size 1) and one softPlus layer that approximates the ReLU function smoothly [28]. Its input is the noise level of the input images and the output is the hyper-parameters in (5), i.e., $\{P^k\}_{k=1}^K = \{\alpha^k, \lambda^k, \rho^k, \eta_n^k \ (n = 1, 2, 3)\}_{k=1}^K$.

In the end of the network, the final representation coefficients are obtained by the last CDR module, and the clean image is reconstructed by applying the convolution layer corresponding to $\mathbf{D}$ to the final coefficients, which is consistent with the reconstruction of signals in traditional dictionary learning.

3.2.1. Convolutional Dictionary Reconstruction module

The update of $\mathbf{X}$ and $\mathbf{D}$ is unfolded as a CDR module. As mentioned in Section 3.1, the dictionary and its transpose are generalized as two convolutional layers. The structure of the CDR module, as shown in Fig. 2, is developed based on equation (5a).

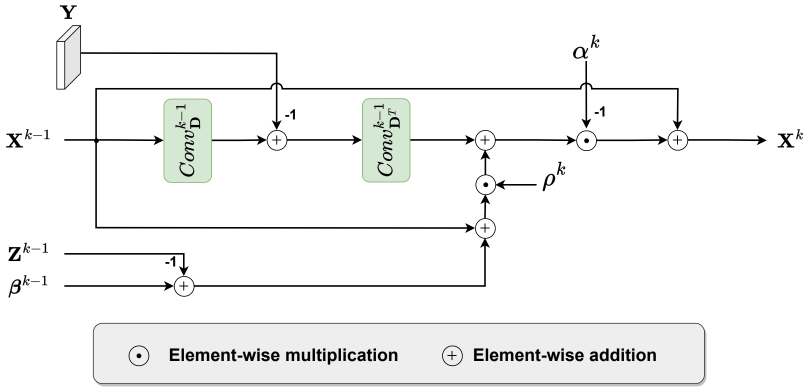


Figure 2: The Convolutional Dictionary Reconstruction (CDR) module.

The output $\mathbf{X}^k$ of this module is based on $\mathbf{X}^{k-1} \in \mathbb{R}^{W\times H\times C}$, $\mathbf{Z}^{k-1} \in \mathbb{R}^{W\times H\times C}$, and $\boldsymbol{\beta}^{k-1} \in \mathbb{R}^{W\times H\times C}$ from the previous stage, where W, H and C are the dimensions and number of channels, respectively. $\mathbf{Y} \in \mathbb{R}^{W\times H\times C_{in}}$ is the input image with additive white Gaussian noise of standard deviation, i.e., noise level σ, and C_{in} is the number of channels of the input image. The convolutional layer $Conv_{\mathbf{D}}^{k-1}$ (kernel size 3) corresponds to the dictionary of the $(k-1)$-th stage, and it converts C channels to C_{in} channels ($C_{in}=1$ for gray images or $C_{in}=3$ for color images). The convolutional layer $\mathbf{C}onv_{D^T}^{k-1}$ (kernel size 3) denotes $\mathbf{D}^T$ of the $(k-1)$-th stage, and the number of channels is converted from C_{in} to C via this layer. In the proposed method, we set $C=16$ to get deep features.

3.2.2. Interstage Information Fusion Transformer Module

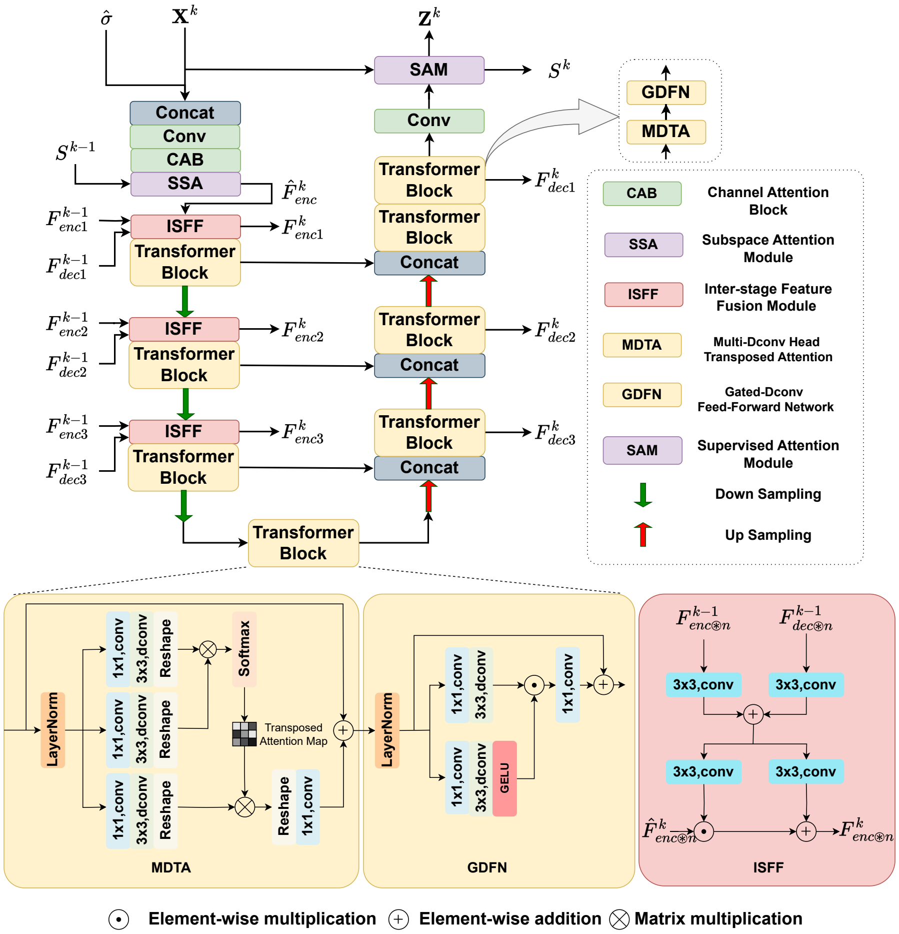


Figure 3: The Interstage Information Fusion Transformer (IIFT) module.

The subproblem $Solve_{\mathbf{Z}}$ in equation (4) is regarded as a denoising problem and the IIFT module is developed to address it. In the update of $\mathbf{Z}$, the prior of $\mathbf{Z}$ is learned implicitly from data. IIFT has an encoder-decoder structure composed of three down-sampling and up-sampling layers. To reduce the loss of information between different stages, we introduce an Inter-Stage Feature Fusion (ISFF) module [34]. The detailed structure of the IIFT module is shown in Fig. 3, including the structures of its core modules Transformer Block and ISFF.

The noise level σ is replicated and extended to $\hat{\sigma}$, a tensor of the same dimension as $\mathbf{X}^k$. The concatenation of $\mathbf{X}^k$ and $\hat{\sigma}$ is used as the input of IIFT, to improve the adaptability to noise levels [28]. A convolutional layer (kernel size 3) is then employed to convert the channels of the input from $C + 1$ to C, and a Channel Attention Block (CAB) is utilized for feature reinforcement. To refine the features obtained in the previous state, a Supervised Attention Module (SAM) [57] is used to obtain the attention map of the current stage, i.e., S^k, and the attention map of the previous stage S^{k-1} is injected into a Subspace Attention (SSA) module [58] of the current stage.

In the three down-sampling steps, each step uses a down-sampling convolutional layer, an ISFF module [34] and a Transformer Block consisting of a Multi-Dconv head Transposed Attention (MDTA) and a Gated-Dconv Feed-forward Network (GDFN) module [35]. The corresponding up-sampling steps use an up-sampling convolutional layer and a Transformer Block same as those in the down-sampling steps.

(1) ISFF: The ISFF module aims to fuse cross-stage information to reduce the loss of information between stages, which can fuse the encoder and decoder features obtained in the previous stage. The formulation of ISFF [34] is defined as

$$\begin{cases} \mathbf{H}_n^{k-1} = \mathrm{Conv}_1\left(\mathbf{F}_{enc\circledast n}^{k-1}\right) + \mathrm{Conv}_2\left(\mathbf{F}_{dec\circledast n}^{k-1}\right), \\ \mathbf{F}_{enc\circledast n}^{k} = \hat{\mathbf{F}}_{enc\circledast n}^{k} \odot \mathrm{Conv}_3\left(\mathbf{H}_n^{k-1}\right) + \mathrm{Conv}_4\left(\mathbf{H}_n^{k-1}\right), \end{cases} \tag{6}$$

where $\mathrm{Conv}_i(i = 1, 2, 3, ...)$ denotes the convolution operation, $\odot$ denotes the element-wise multiplication, and $\hat{\mathbf{F}}_{enc\circledast n}^{k}$ represents the input feature map of

ISFF in the current stage. $\mathbf{F}_{enc\circledast n}^{k-1} \in \mathbb{R}^{\frac{W}{2^n} \times \frac{H}{2^n} \times 2^n C}$ and $\mathbf{F}_{dec\circledast n}^{k-1} \in \mathbb{R}^{\frac{W}{2^n} \times \frac{H}{2^n} \times 2^n C}$ denote the encoder and decoder features of the n-th ($n = 0, 1, 2$) scale of the $k-1$ stage, respectively. The output of ISFF $\mathbf{F}_{enc\circledast n}^{k}$ is injected into the Transformer Block.

(2) Transformer Block: The Transformer Block consists of MDTA and GDFN modules, where MDTA focuses on global features and GDFN aims to obtain local information of spatially neighboring pixels. This block can get output feature maps with global context information and local structures of the input feature maps.

The formulation of MDTA is defined as [35]

$$\begin{aligned} \hat{\mathbf{Z}}_M &= \mathrm{Conv}^p \,\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) + \mathbf{F}, \\ \mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) &= \mathbf{V} \cdot \mathrm{Softmax}(\mathbf{K} \odot \mathbf{Q}/\alpha), \end{aligned} \tag{7}$$

where the query ($\mathbf{Q}$), key ($\mathbf{K}$) and value ($\mathbf{V}$) projections are generated from the input feature $\mathbf{F}$, i.e., $\mathbf{Q} = \mathrm{Conv}_Q^d(\mathrm{Conv}_Q^p(\mathbf{F}))$, $\mathbf{K} = \mathrm{Conv}_K^d(\mathrm{Conv}_K^p(\mathbf{F}))$, $\mathbf{V} = \mathrm{Conv}_V^d(\mathrm{Conv}_V^p(\mathbf{F}))$. $\mathrm{Conv}_{(.)}^p$ is the 1×1 point-wise convolution and $\mathrm{Conv}_{(.)}^d$ is the 3×3 depth-wise convolution to enhance local features. $\mathrm{Conv}_Q^{(.)}$, $\mathrm{Conv}_K^{(.)}$, and $\mathrm{Conv}_V^{(.)}$ denote the convolution layers to obtain the query, key and value projections. α is the learning parameter to constrain the element-wise multiplication for stability. $\hat{\mathbf{Z}}_M$ is the output feature map through generating a global attention map of cross-channels, which contains enriching context information. $\hat{\mathbf{Z}}_M$ is then fed into GDFN to learn the local structural information of the feature map.

The GDFN module is defined as [35]

$$\begin{aligned} \hat{\mathbf{Z}}_G =& \mathrm{Conv}_5^p \,\mathrm{Gating}(\hat{\mathbf{Z}}_M) + \hat{\mathbf{Z}}_M, \\ \mathrm{Gating}(\hat{\mathbf{Z}}_M) =& \mathrm{GELU}\left(\mathrm{Conv}_6^d \,\mathrm{Conv}_6^p(\mathrm{LN}(\hat{\mathbf{Z}}_M))\right) \\ & \odot \mathrm{Conv}_7^d \,\mathrm{Conv}_7^p(\mathrm{LN}(\hat{\mathbf{Z}}_M)), \end{aligned} \tag{8}$$

where GELU represents a non-linear activation layer based on Gaussian error linear unit, and LN is the layer normalization. $\hat{\mathbf{Z}}_G$ is the output feature map of GDFN.

By integrating the ISFF module and the Transformer Block, we propose the IIFT module to get the global feature map and capture long-range pixel interactions. The IIFT module is utilized to replace $\mathcal{S}(.)$ in (5), and embedded into the iterative ADMM algorithm and served as a denoiser. In this way, the proposed unfolding model can make full use of DNN to learn the local structure information and global structure simultaneously.

3.2.3. Lagrangian Multipliers Update Module

The LMU module is used to update the Lagrangian multiplier $\boldsymbol{\beta}$ in equation (5c). The input of this module in the k-th stage is the output of the LMU module in the previous stage and the output of the CDR and IIFT modules in the current stage: $\boldsymbol{\beta}^{k-1}, \mathbf{X}^k$, and $\mathbf{Z}^k$. The output of LMU module is defined as

$$\boldsymbol{\beta}^k = \eta_1^k \boldsymbol{\beta}^{k-1} + \eta_2^k \mathbf{X}^k - \eta_3^k \mathbf{Z}^k, \tag{9}$$

where $\boldsymbol{\beta}^{k-1}$ denotes the Lagrangian multiplier of stage $k-1$. The parameters η_n^k $(n = 1, 2, 3)$ are obtained via the HypaNet [28], as mentioned in Fig. 1 of Section 3.2.

3.2.4. Loss function

The unfolded network is trained in a supervised manner rather than unsupervised as in traditional dictionary learning. Given the ground-truth image $\mathbf{Y}^{gt}$, the noisy image $\mathbf{Y}$, a training set Γ consisting of pairs of $\mathbf{Y}^{gt}$ and $\mathbf{Y}$ can be constructed. The averaged ℓ_1 loss is used as the loss function, which is defined as

$$E(\boldsymbol{\Theta}) = \frac{1}{|\Gamma|} \sum_{(\mathbf{Y}, \mathbf{Y}^{gt}) \in \Gamma} \left\| \hat{\mathbf{Y}}(\mathbf{Y}, \boldsymbol{\Theta}) - \mathbf{Y}^{gt} \right\|_1, \tag{10}$$

where $\hat{\mathbf{Y}}$ is the output of the TDU-DLNet model, and $\boldsymbol{\Theta} = \{Conv_{\mathbf{D}}^k, Conv_{\mathbf{D}^T}^k, \alpha^k, \rho_k, \eta_n^k \ (n = 1, 2, 3)\}_{k=1}^K$ denotes the set of parameters to be learned in the model.

3.2.5. Comparison with related deep unfolding methods

Table 1 compares the components of the proposed method with those related deep unfolding methods in terms of underlying formulation, underlying iterative

Table 1: Comparison with related deep unfolding methods

Methods	Underlying Formulation	Underlying Iterative Algorithm	Neural Networks Embedded
DGUNet [34]	Image restoration with an implicit prior on the image	Proximal Gradient Descent (PGD)	CNN-based network with ISFF
LKSVD [27]	Dictionary learning with an ℓ_1-norm prior on representation coefficients	Iterative Soft-Thresholding Algorithm (ISTA)	Multi-Layer Perceptron (MLP) network
DCDicL [28]	Convolutional dictionary learning with two implicit priors: a prior on the dictionary and a prior on representation coefficients	Half Quadratic Splitting (HQS)	CNN-based network
TDU-DLNet	Dictionary learning with an implicit prior on representation coefficients	Alternating Direction Method of Multipliers (ADMM)	Transformer-based network with ISFF

algorithm, and neural networks embedded. From this table, it is clear that the proposed method is radically different from existing deep unfolding methods. The method DGUNet [34] is based on a formulation for image restoration by considering a prior on the image directly, while the proposed method is a generalization of traditional dictionary learning and applies a prior to the representation coefficients based on a learned dictionary. As compared with the deep unfolding methods for dictionary learning (LKSVD [27] and DCDicL [28]), the proposed method is based on a general formulation for dictionary learning with only one implicit prior on representation coefficients, and its architecture is developed by unrolling the iterative steps of ADMM and embedding a transformer-based network with ISFF.

Deep unfolding methods attempt to integrate the good interpretability of model-based methods and powerful learning ability of deep learning methods. In these methods, iterative algorithms based on traditional methods are unrolled as approximate neural networks, and the parameters of the unfolded model are learned with deep neural networks in an end-to-end fashion. Our proposed model follows the framework of deep unfolding, and the overall architecture is constructed by unrolling the iterative steps generalized from ADMM. Due to approximations used in the unrolling process, the interpretability of the proposed model is not as good as the original dictionary learning model, but it is better than that of pure deep learning networks.

4. Experiment

As the formulation on which the proposed unfolding network is based is a general one without considering the degradation process of images, the proposed network can be extended to other image reconstruction problems with the same input and output size, such as image restoration and image deblurring, when corresponding data sets are available. In this section, we take image denoising as an example and perform experiments to demonstrate the performance of the proposed model as compared with existing related methods. Ablation studies on the number of stages and different components of the proposed model are also given.

4.1. Training details

Following the experiments in [28], we utilized the combination of datasets BSD400 [59], DIV2K [60] and WED [61] for training. These datasets contain pairs of clean images and noisy images with additive white Gaussian noise. Patches of size 128×128 randomly extracted from clean and noisy image pairs of the datasets were used as training data. The noise level σ, i.e., the standard deviation of additive white Gaussian noise, is a random number from 15 to 50.

We used the Adam optimizer to update the learnable parameters and the maximum epoch was 200. In order to speed up the training, the learning rate r was set as $1e-3$ initially and decayed to $1e-4$ by a factor of 0.5 for the first 50 epochs, and r decayed to $1e-6$ by a factor of 0.8 for the last 150 epochs. The batch size was set as 4 for $2e5$ iterations. We set the number of stages

$K = 5$. From level-1 to level-4 of the IIFT module, the number of channels of the Transformer blocks are 16, 32, 64, and 128, respectively. The model training was performed on a Nvidia GeForce RTX 3090 GPU[1].

4.2. Ablation studies

4.2.1. Selection of K

To investigate the impact of the stage number K, different settings of K were tested with all other parameters being fixed. The results on CBSD68 [59] are presented in Fig. 4, where the horizontal axis refers the inference time using the trained model and the vertical axis refers to the Peak Signal-to-Noise Ratio (PSNR) of the denoised images.

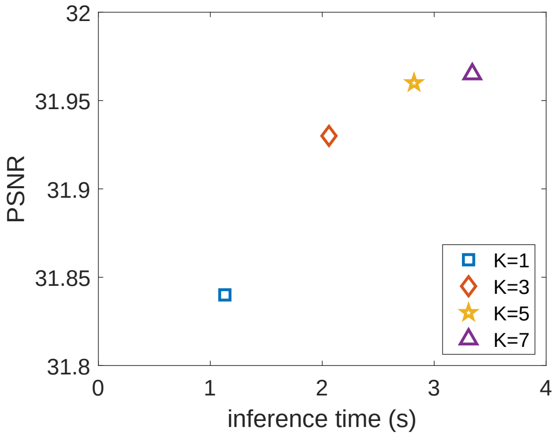


Figure 4: PSNR (dB) results with different settings of stage number K.

It can be seen that the values of PSNR rise with the increase of K. The PSNR value at $K = 7$ is slightly higher than that at $K = 5$, while the former requires more inference time. In order to balance the performance and time consumption, we set $K = 5$ in the following experiments.

[1] The code of the proposed algorithm is available at `https://github.com/wuhu1010/TDUDLNet`.

4.2.2. Ablation study with respect to IIFT

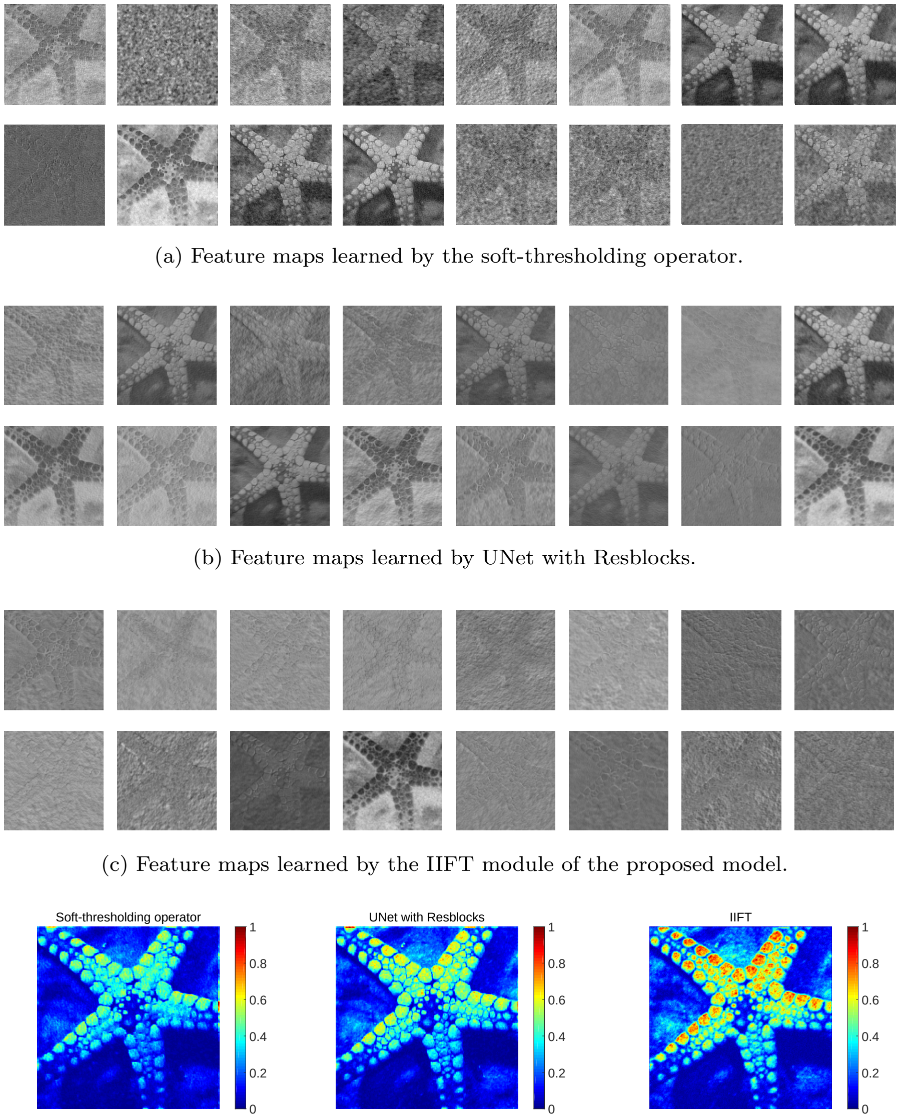
To validate the effect of the IIFT module of the proposed model, we replaced the IIFT module as the soft-thresholding operator or UNet with Resblocks [28]. In [31], the soft-thresholding operator is used to obtain sparse coefficients. UNet with Resblocks [28] is widely used network for image denoising, and the four blocks corresponding to the up-down sampling have 16, 32, 64, 128 channels same as the settings of the proposed model. To investigate the impact of the ISFF module used in IIFT, we tested the proposed model using IFFT without ISFF, the model using UNet with Resblocks with ISFF embedded, and IIFT with the feature fusion (FF) module [52]. The results of these ablation studies on CBSD68 are summarized in Table 2.

Table 2: Results of ablation studies related to IIFT.

Employment of ISFF	Modules used to address $Solve_{\mathbf{Z}}$	PSNR (in dB)			Params
		$\sigma = 15$	$\sigma = 25$	$\sigma = 50$	
Without ISFF	Soft-thresholding operator	32.36	29.97	26.58	12.03K
	UNet with Resblocks	34.20	31.62	28.45	2.05M
	IIFT *w/o* ISFF	34.39	31.77	28.59	3.06M
	IIFT with FF [52]	34.43	31.83	28.66	3.08M
With ISFF	UNet with Resblocks	34.29	31.70	28.52	2.83M
	IIFT (Ours)	**34.54**	**31.94**	**28.77**	3.26M

Though ISFF is not used, the proposed model using IIFT outperforms the models using soft thresholding or UNet with Resblocks. When ISFF is embedded, both the proposed model using IIFT and UNet with Resblocks achieve better results than their versions without ISFF. When the FF module [52] is used in IIFT, the denoising performance is better than IIFT without any feature fusion modules but worse than IIFT with ISFF. This demonstrates the effectiveness of the proposed IIFT module and the employment of ISFF.

To compare the prior features learned by the soft-thresholding operator, UNet with Resblocks, and the IIFT module of the proposed model intuitively, we used the image 04 in Set12 [62] as the test image and visualized its feature maps learned by different methods in Fig. 5. It can be seen that feature maps learned by soft-thresholding and UNet with Resblocks emphasize subtle textures and details, while those by IIFT present the general structure without emphasizing



(a) Feature maps learned by the soft-thresholding operator.

(b) Feature maps learned by UNet with Resblocks.

(c) Feature maps learned by the IIFT module of the proposed model.

(d) Visualization of the variances of the feature maps learned by different models.

Figure 5: Visualization of feature maps and their variances learned by different models.

details. To quantify the differences of the feature maps in terms of diversity, we calculate the variances of feature maps along channels, i.e., the variances of the pixel values at the same location of the feature maps obtained by one model. The variances of the feature maps obtained by different models are visualized in Fig. 5-(d) and the average values of the variances for soft-thresholding operator, UNet with Resblock, and IIFT are 0.033, 0.051, and 0.084, respectively. It can be seen that the variances of feature maps obtained by IIFT are larger in general than those by the other two models. This demonstrates that the features learned by the IIFT module exhibit greater diversity as compared with the soft-thresholding operator and UNet with Resblocks.

4.2.3. Ablation study with respect to CDR and LMU

To demonstrate the benefits of CDR and LMU, we performed two ablation experiments related to these two modules. In the first ablation setting, we eliminated the CDR and LMU modules in each stage of the proposed deep unfolding architecture, and only retained the IIFT module. In the second ablation setting, we fixed the parameters of the CDR and LMU modules as the initialized parameters and only updated the parameters of the remaining modules. These two variants were trained in the same way as the proposed model. The denoising results on CBSD68 and BSD68 [59] are presented in Table 3.

Table 3: Results of ablation studies related to CDR and LMU.

Datasets	Models	PSNR (in dB)		
		$\sigma = 15$	$\sigma = 25$	$\sigma = 50$
CBSD68	Only IIFT	34.40	31.82	28.65
	Fixed CDR and LMU	34.05	31.46	28.35
	Proposed	**35.54**	**31.94**	**28.77**
BSD68	Only IIFT	31.79	29.47	26.61
	Fixed CDR and LMU	31.43	29.11	26.25
	Proposed	**32.06**	**29.65**	**26.77**

It can be seen that the complete version of the proposed model outperforms the other two variants. This is due to the information interaction between the modules and the various stages in the architecture of the proposed model. Though the IIFT module takes most of the learnable parameters, it only plays a part in the whole architecture.

4.3. Comparison with other methods

In this section, we compare the proposed TDU-DLNet with several state-of-the-art image denoising methods on standard grayscale and color image datasets. Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) [63] are used to evaluate the denosing results of the models. The quantitative results of the competitive models were cited directly from the original papers.

4.3.1. Color image denoising

For color image denoising, we adopt the test datasets including CBSD68 [59], Kodak24 [64], McMaster [65] and Urban100, and compared the proposed model with classical deep learning methods (DnCNN [62], BRDNet [66], and RPCNN [67]), deep unfolding methods (DCDicL [28] and DGUNet [34]) and the state-of-the-art models (Restormer [35] and CODE [68]). Among these methods, BRDNet and RPCNN learn a specific model for each noise level, while the other methods including the proposed method learn a general model for different noise levels. The denoising results for colored images are reported in Table 4, and samples of denoised images and residuals obtained by different methods are presented in Figs. 6-9.

From Table 4, it can be seen the deep unfolding model DCDicL gets better performance than the deep learning models except for Restormer. The proposed TDU-DLNet achieves the best results in terms of PSNR on the CBSD68, Kodak24 and McMaster datasets for most noise levels. For the Urban100 dataset, Restormer obtains the best results, and the proposed model outperforms all the other deep learning methods. The Urban100 dataset contains high-resolution images with fine-scale repetitive structures and textures. Denoising on this dataset is more challenging than the other datasets. Though the proposed

Table 4: Denoising results for color images in PSNR(dB)/SSIM(%).

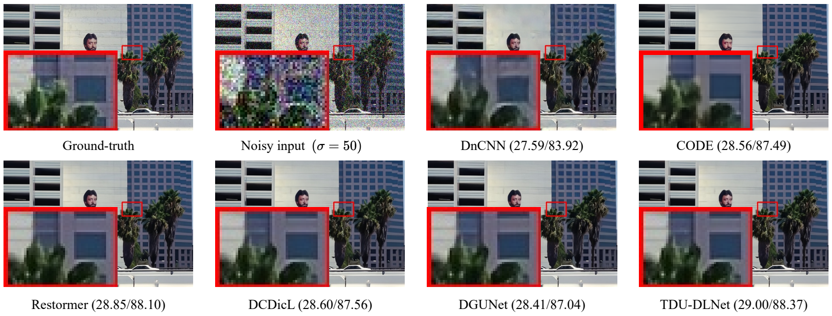
Datasets	σ	DnCNN [62]	BRDNet [66]	RPCNN [67]	Restormer [35]	CODE [68]	DCDicL [28]	DGUNet [34]	TDU-DLNet
CBSD68	15	33.90/92.91	34.10/92.91	-	34.40/93.55	34.33/**93.80**	34.36/93.48	34.20/93.32	**34.54**/93.58
	25	31.24/88.31	31.43/88.47	31.24/88.80	31.80/89.44	31.69/**89.70**	31.75/89.30	31.60/89.07	**31.94**/89.49
	50	27.95/78.98	28.16/79.42	28.06/79.90	28.63/81.37	28.47/81.34	28.57/81.07	28.40/80.68	**28.77/81.45**
Kodak24	15	34.47/91.98	34.88/92.49	-	35.33/93.02	35.32/**93.12**	35.38/93.00	35.03/92.71	**35.40**/93.04
	25	32.02/87.64	32.41/88.56	32.34/88.40	32.92/89.36	32.88/**89.39**	32.97/89.28	32.63/88.90	**33.00/89.39**
	50	28.83/79.11	29.22/80.40	29.25/80.50	29.88/**82.39**	29.82/82.02	**29.96**/82.19	29.55/81.52	29.94/82.38
McMaster	15	33.45/90.36	35.08/92.69	-	35.56/93.41	35.38/**93.51**	35.50/93.35	35.14/92.92	**35.61**/93.48
	25	31.52/86.95	32.75/89.43	32.33/89.00	33.33/90.60	33.11/90.60	33.26/90.48	32.93/89.96	**33.39/90.74**
	50	28.61/79.86	29.52/82.65	29.35/82.60	30.30/85.20	30.03/84.68	30.22/84.94	29.87/84.11	**30.34/85.38**
Urban100	15	32.98/93.15	34.42/94.62	-	**35.06/95.15**	-	34.90/95.11	34.62/94.20	34.94/95.12
	25	30.81/90.15	31.99/91.94	31.81/91.90	**32.91/93.08**	-	32.77/93.00	32.43/91.15	32.77/92.98
	50	27.59/83.31	28.56/85.77	28.62/86.20	**30.02/88.94**	-	29.88/88.84	29.48/87.52	29.78/88.60
Average		31.12/86.89	31.88/88.28	30.38/85.92	32.51/89.63	32.34/88.69	32.46/89.51	32.16/88.84	**32.54/89.64**

method also uses long-range dependency as in Restormer, the numbers of channels of the MDTA module in Transformer blocks of the proposed model are only one-third of the settings in Restormer. From level-1 to level-4 of the encoder-decoder architecture, the numbers of channels of Transformer blocks in the proposed method and Restormer are [16, 32, 64, 128] and [48, 96, 192, 384], respectively. With a reduced number of parameters, the capability of the model can be compromised, which may lead to performance degradation in denoising the challenging images in Urban100.

In terms of SSIM, CODE obtains the best results for the cases of lower noise levels. To compare the overall performance of the models, the average results over all datasets and all noise levels are presented in the last row of the table. The proposed model achieves the best average results for both PSNR and SSIM. From Figs. 6-9, it can be observed that the proposed model can restore edge details corrupted by noise and obtain clean images of good quality.

4.3.2. Grayscale image denoising

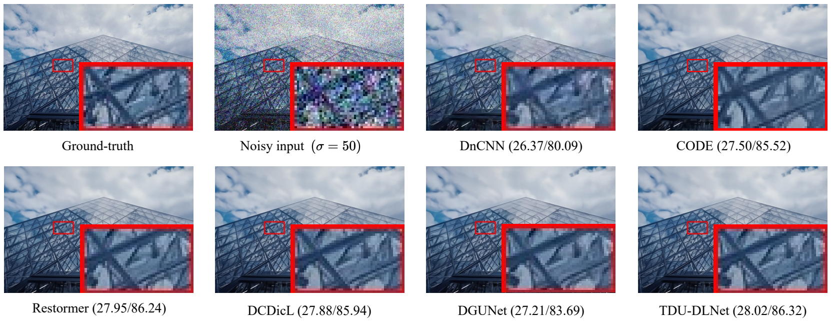
For grayscale image denoising, we used the testing datasets including Set12 [62], BSD68 [59] and Urban100 [69], and compared the proposed model with



Ground-truth | Noisy input ($\sigma = 50$) | DnCNN (27.59/83.92) | CODE (28.56/87.49)

Restormer (28.85/88.10) | DCDicL (28.60/87.56) | DGUNet (28.41/87.04) | TDU-DLNet (29.00/88.37)

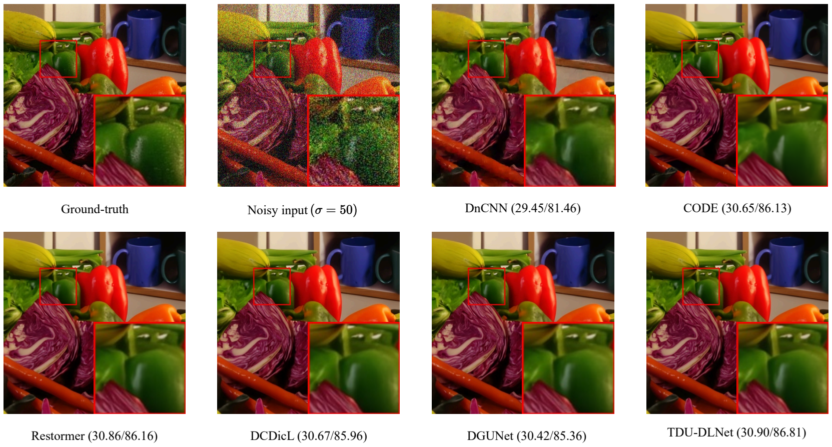
Figure 6: Denoising results on image 119082 in CBSD68.



Ground-truth | Noisy input ($\sigma = 50$) | DnCNN (26.37/80.09) | CODE (27.50/85.52)

Restormer (27.95/86.24) | DCDicL (27.88/85.94) | DGUNet (27.21/83.69) | TDU-DLNet (28.02/86.32)

Figure 7: Denoising results on image 223061 in CBSD68.

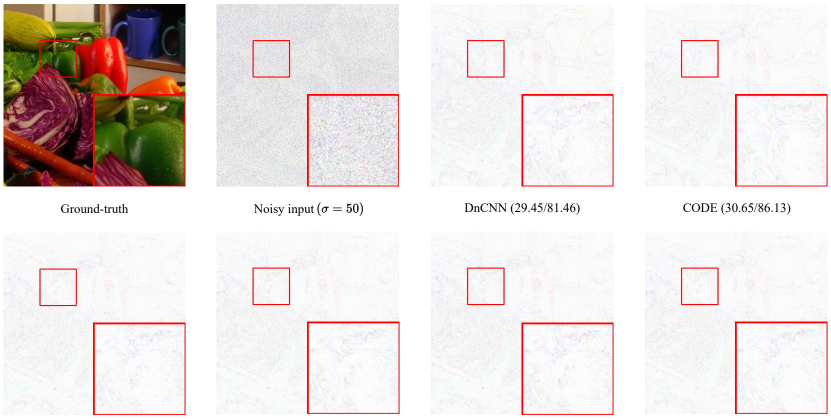


Ground-truth | Noisy input ($\sigma = 50$) | DnCNN (29.45/81.46) | CODE (30.65/86.13)

Restormer (30.86/86.16) | DCDicL (30.67/85.96) | DGUNet (30.42/85.36) | TDU-DLNet (30.90/86.81)

Figure 8: Denoising results on image 10 in McMaster.

classical deep learning methods (DnCNN [62] and FFDNet [17]), deep unfolding methods (DCDicL [28] and DGUNet [34]) and the state-of-the-art models (Restormer [35], CODE [68], and NERD [70]). The denosing results for grayscale



Ground-truth

Noisy input ($\sigma = 50$)

DnCNN (29.45/81.46)

CODE (30.65/86.13)

Restormer (30.86/86.16)

DCDicL (30.67/85.96)

DGUNet (30.42/85.36)

TDU-DLNet (30.90/86.81)

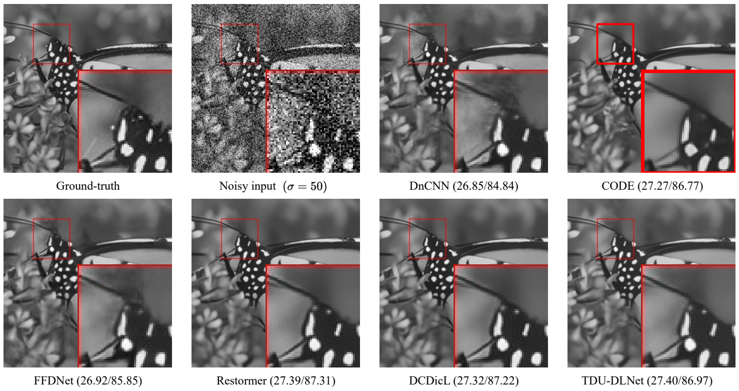
Figure 9: Denoising residuals on image 10 in McMaster.

images are summarized in Table 5. Figs. 10-13 present the denoised results and residuals of different models.

Table 5: Denoising results for grayscale images in PSNR(dB)/SSIM(%).

Datasets	σ	DnCNN [62]	FFDNet [17]	Restormer [35]	NERD [70]	CODE [68]	DCDicL [28]	DGUNet [34]	TDU-DLNet
Set12	15	32.86/90.24	32.75/90.24	**33.35**/91.15	33.20/91.18	33.33/**91.42**	33.34/91.15	33.14/90.62	33.30/90.97
	25	30.44/86.17	30.43/86.31	**31.04**/87.53	30.84/87.36	31.01/**87.68**	31.03/87.48	30.82/87.02	31.00/87.33
	50	27.18/78.28	27.32/78.99	**28.01**/81.21	27.72/80.75	27.93/80.82	28.00/**81.22**	27.80/80.54	27.96/80.89
BSD68	15	31.73/89.07	31.63/89.02	31.97/89.64	31.91/89.75	31.96/**90.15**	31.95/89.57	31.78/89.36	**32.06**/89.60
	25	29.23/82.79	29.19/82.88	29.54/83.92	29.43/83.70	29.51/**84.40**	29.52/83.79	29.36/83.42	**29.65**/83.85
	50	26.23/71.89	26.29/72.39	26.66/**74.22**	26.49/73.89	26.58/74.09	26.63/73.95	26.47/73.03	**26.77**/74.06
Urban100	15	32.64/92.46	32.40/92.65	**33.70/93.97**	33.48/93.41	-	33.59/93.88	33.30/93.44	33.35/93.66
	25	29.95/87.81	29.90/89.79	**31.41/91.19**	31.03/90.13	-	31.30/91.08	30.81/90.43	31.00/90.67
	50	26.26/78.56	26.50/80.47	**28.34/85.65**	27.62/82.92	-	28.24/85.49	27.39/84.09	27.79/84.38
Average		29.61/84.14	29.60/84.75	**30.45/86.50**	30.19/85.90	30.06/84.76	30.40/86.40	30.10/85.77	30.32/86.16

It can be seen from Table 5 that the proposed algorithm obtains the best PSNR results on the BSD68 among all competing methods. For the Set12 and Urban100 datasets, the proposed model outperforms the deep learning methods except for Restormer and the deep unfolding method DCDicL in terms of PSNR. Restormer and DCDicL obtain slightly better results than the proposed method, but the numbers of training parameters of these two models are a few times of that of the proposed model, as will be demonstrated later. NERD and CODE


Ground-truth

Noisy input ($\sigma = 50$)

DnCNN (26.85/84.84)

CODE (27.27/86.77)

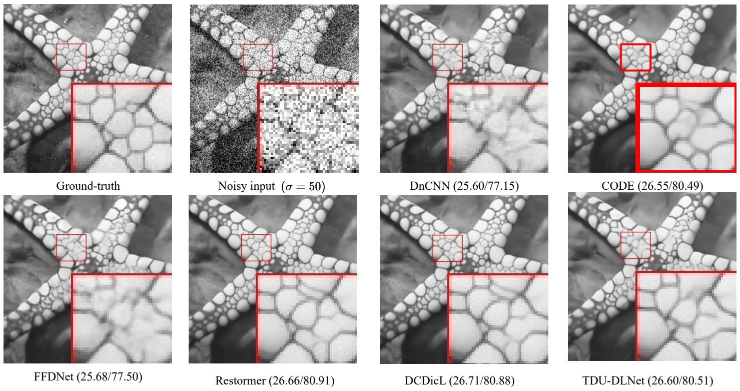
FFDNet (26.92/85.85)

Restormer (27.39/87.31)

DCDicL (27.32/87.22)

TDU-DLNet (27.40/86.97)

Figure 10: Denoising results on image 05 in Set12.


Ground-truth

Noisy input ($\sigma = 50$)

DnCNN (25.60/77.15)

CODE (26.55/80.49)

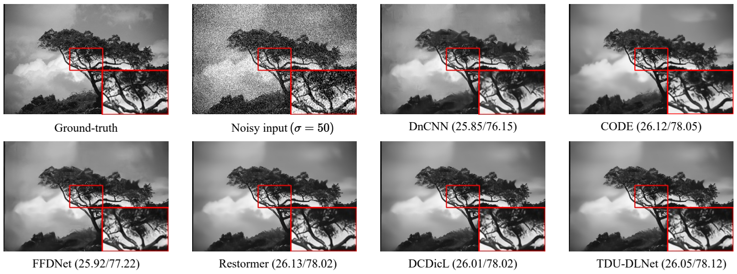
FFDNet (25.68/77.50)

Restormer (26.66/80.91)

DCDicL (26.71/80.88)

TDU-DLNet (26.60/80.51)

Figure 11: Denoising results on image 04 in Set12.


Ground-truth

Noisy input ($\sigma = 50$)

DnCNN (25.85/76.15)

CODE (26.12/78.05)

FFDNet (25.92/77.22)

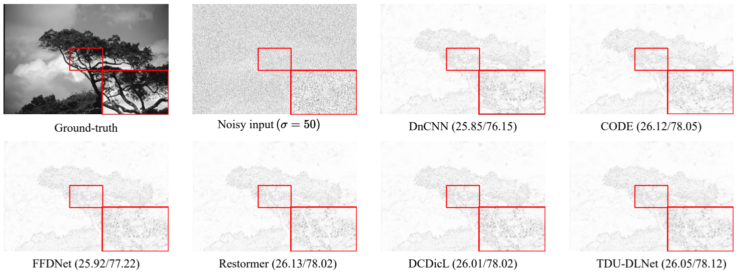
Restormer (26.13/78.02)

DCDicL (26.01/78.02)

TDU-DLNet (26.05/78.12)

Figure 12: Denoising results on image 20 in BSD68.

perform better for lower noise levels in terms of SSIM. Figs. 10-13 show the proposed method can restore clean images and get better performance than the compared deep learning methods.



Ground-truth

Noisy input ($\sigma = 50$)

DnCNN (25.85/76.15)

CODE (26.12/78.05)

FFDNet (25.92/77.22)

Restormer (26.13/78.02)

DCDicL (26.01/78.02)

TDU-DLNet (26.05/78.12)

Figure 13: Denoising residuals on image 20 in BSD68.

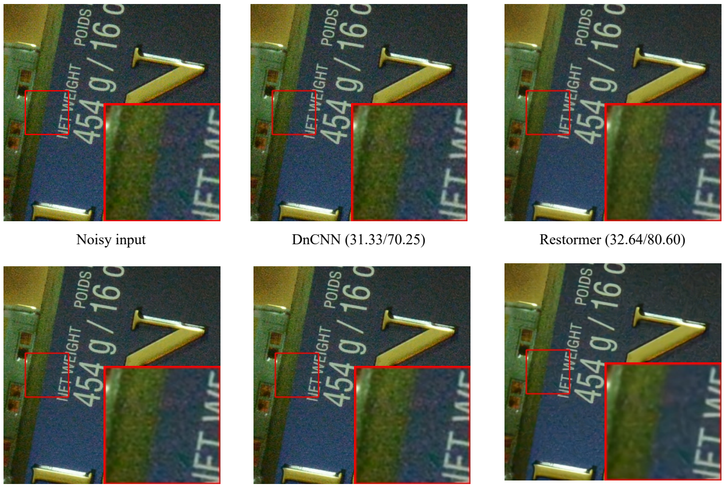
4.3.3. Generalization to high-level and real noise

As mentioned in Section 4.1, the proposed model is trained based on synthetic noisy images with additive white Gaussian noise, and the noise level is from 15 to 50. To test the generalization ability of the model to images with higher-level noise and real noisy images, we applied the pre-trained models to images with noise level $\sigma = 75$ and real noisy images in [71]. The real noisy image dataset in [71] contains 15 color noisy images of size 512×512 and the corresponding mean images averaged from 500 shots of the same scene under controlled indoor environment. The mean images can be used as the "ground-truth" to compute quantitative results of denoising. The results on noisy images with $\sigma = 75$ and real noisy images are presented in Table 6. The denoised images of a real image sample are presented in Fig. 14.

Table 6: Denoising results for images with high-level noise ($\sigma = 75$) and real noisy images (PSNR(dB)/SSIM(%)).

Image Type	Datasets	DnCNN [62]	Restormer [35]	DCDicL [28]	DGUNet[34]	TDU-DLNet
Color Images	CBSD68	24.50/59.49	26.97/75.67	26.88/74.69	26.89/75.37	**27.12/75.89**
	Kodak24	25.00/57.39	28.23/**77.52**	28.02/75.82	28.15/77.26	**28.28**/77.20
	McMaster	25.09/59.02	28.54/81.13	28.32/80.09	28.46/80.91	**28.60/81.29**
	Urban100	24.18/64.94	**28.29/85.56**	27.69/83.25	28.20/85.37	27.98/84.54
Grayscale Images	Set12	18.73/29.73	**26.23**/76.51	26.07/75.47	25.99/75.09	26.15/**75.79**
	BSD68	18.73/30.45	25.16/**68.17**	25.17/67.45	25.09/66.91	**25.24**/67.88
	Urban100	18.99/39.12	**26.51/81.05**	25.73/78.64	25.65/78.30	25.81/78.93
Real Noisy Images	[71]	33.86/86.36	36.31/**94.02**	36.35/93.16	35.82/91.91	**36.56**/93.64

The proposed method outperforms DnCNN, DCDicL, and DGUNet, and achieves comparable results as compared with Restormer. The denoised image shown in Fig. 14 illustrates the good generalization of the proposed model to real noisy images.



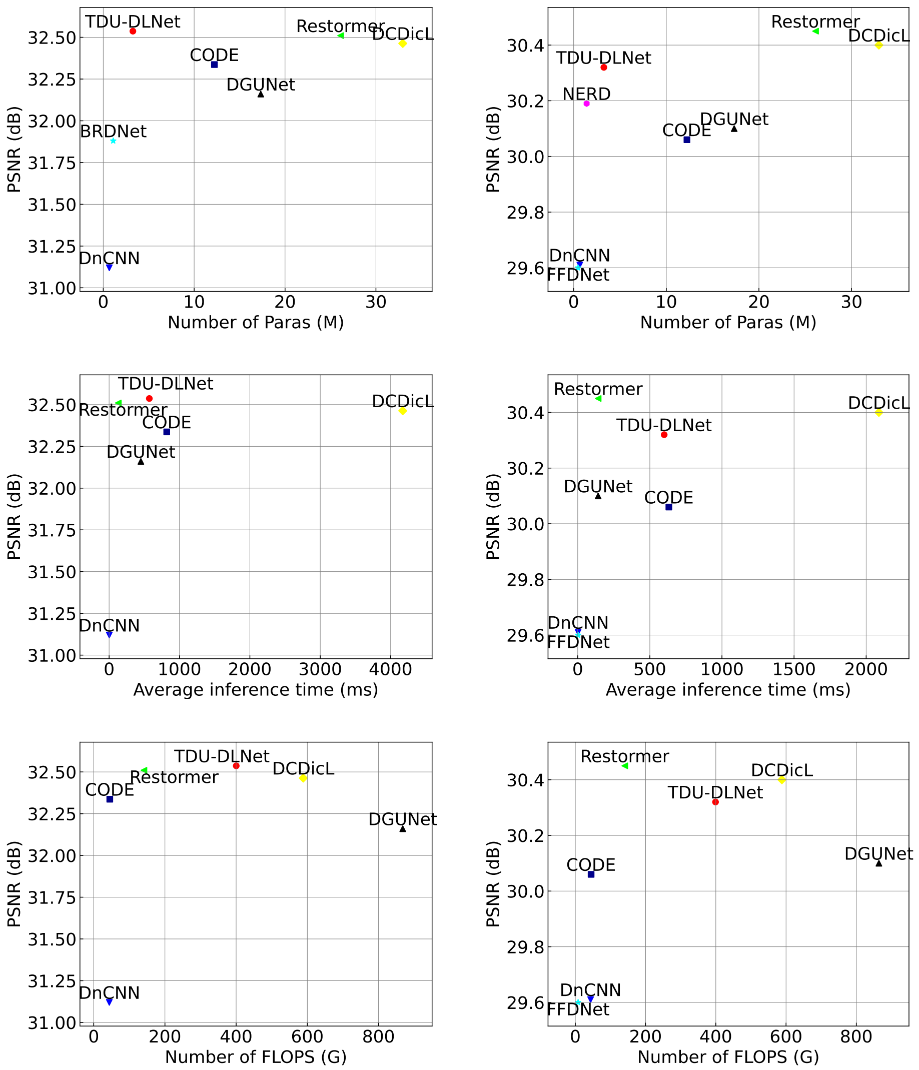
Noisy input DnCNN (31.33/70.25) Restormer (32.64/80.60)

DcDicL (32.30/78.31) DGUNet (32.07/76.52) TDU-DLNet (33.75/86.70)

Figure 14: Denoising results on image 13 in real noisy image dataset [71].

4.3.4. Model Complexity

The number of training parameters, inference time, and the number of FLOPs versus the denoising results of different models are presented in Fig. 15. The PSNR results shown in the figure are the average results of the models presented in Tables 4-5. The number of FLOPS is computed on image size 256 × 256. The proposed model has fewer training parameters than CODE, DGUNet, Restormer and DCDicL, but achieves competitive results. Although the numbers of training parameters of BRDNet, DnCNN, FFDNet, and NERD are smaller than those of other methods, their denoising performance is worse than other methods. The inference time of TDU-DLNet is medium, which is longer than Restormer and DGUNet and shorter than CODE and DCDicL. The deep unfolding models have more FLOPS than deep learning models, and the number of FLOPS of the proposed model is the smallest among the compared deep unfolding models.



(a) Color images (Table 4)

(b) Gray images (Table 5)

Figure 15: The number of training parameters, average inference time, and the number of FLOPS *v.s.* average PSNR results of different methods.

In both the proposed model and the deep learning method Restormer, the Transformer blocks take most of the learnable parameters. As has been mentioned in Section 4.3.1, the numbers of channels of the Transformer blocks in the proposed method are only one-third of those used in Restormer, which reduce the number of parameters dramatically. Despite with a reduced number of channels, the proposed method achieves promising results due to the multi-stage process in the unfolding network. The deep unfolding methods DCDicL and DGUNet have more parameters as the number of channels used in the embedded CNN layers are also high. For example, the number of channels in the convolutional layers in the encoder-decoder architecture of DCDicL are [64, 128, 256, 512].

5. Conclusion

We have proposed a transformer-based deep unfolding framework for dictionary learning (TDU-DLNet). The general model for dictionary learning is employed and an iterative optimization approach is then developed based on ADMM. By unrolling the iterative optimization method, we design a deep unfolding network. A transformer-based module is developed to learn priors of the representation coefficients and an inter-stage information fusion module is introduced to reduce the information loss between different stages of the unfolding network. Extensive experiments for image denosing on several standard datasets have been performed, and the results have demonstrated that the proposed model can obtain better or comparable results, but use fewer parameters, as compared with the state-of-the-art methods. To further improve the performance of the unfolding network, more advanced architectures proposed recently, e.g., Mamba, can be considered in the future.

6. Acknowledgement

This work was partly supported by the National Natural Science Foundation of China (61906087), the Key Research and Development Program of Heilongjiang Province of China (2022ZX01A15), and Cultivation Plan Project of Qingdao Science and Technology Planning Park of China (23-1-5-yqpy-11-qy). The authors wish to thank the Associate Editor and the anonymous reviewers for their contributions to improving the quality of the paper.

References

[1] M. Aharon, M. Elad, A. Bruckstein, K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation, IEEE Transactions on Signal Processing 54 (11) (2006) 4311–4322.

[2] I. Tošić, P. Frossard, Dictionary learning, IEEE Signal Processing Magazine 28 (2) (2011) 27–38.

[3] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T.-W. Lee, T. J. Sejnowski, Dictionary learning algorithms for sparse representation, Neural Computation 15 (2) (2003) 349–396.

[4] W. Dai, T. Xu, W. Wang, Simultaneous codeword optimization (SimCO) for dictionary update and learning, IEEE Transactions on Signal Processing 60 (12) (2012) 6340–6353.

[5] J. Dong, W. Wang, W. Dai, M. D. Plumbley, Z.-F. Han, J. Chambers, Analysis SimCO algorithms for sparse analysis model based dictionary learning, IEEE Transactions on Signal Processing 64 (2) (2016) 417–431.

[6] J. Dong, Z. Han, Y. Zhao, W. Wang, A. Prochazka, J. Chambers, Sparse analysis model based multiplicative noise removal with enhanced regularization, Signal Processing 137 (2017) 160–176.

[7] J. Yang, Z. Wang, Z. Lin, S. Cohen, T. Huang, Coupled dictionary training for image super-resolution, IEEE Transactions on Image Processing 21 (8) (2012) 3467–3478.

[8] B. Li, L. Rencker, J. Dong, Y. Luo, M. D. Plumbley, W. Wang, Sparse analysis model based dictionary learning for signal declipping, IEEE Journal of Selected Topics in Signal Processing 15 (1) (2021) 25–36.

[9] L. Ma, L. Moisan, J. Yu, T. Zeng, A dictionary learning approach for poisson image deblurring, IEEE Transactions on Medical Imaging 32 (7) (2013) 1277–1289.

[10] R. Giryes, M. Elad, Sparsity-based poisson denoising with dictionary learning, IEEE Transactions on Image Processing 23 (12) (2014) 5057–5069.

[11] C. Garcia-Cardona, B. Wohlberg, Convolutional dictionary learning: A comparative review and new algorithms, IEEE Transactions on Computational Imaging 4 (3) (2018) 366–381.

[12] G. Silva, P. Rodriguez, Efficient convolutional dictionary learning using partial update fast iterative shrinkage-thresholding algorithm, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2018, pp. 4674–4678.

[13] H. Bristow, A. Eriksson, S. Lucey, Fast convolutional sparse coding, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 391–398.

[14] I. Y. Chun, J. A. Fessler, Convergent convolutional dictionary learning using adaptive contrast enhancement (CDL-ACE): Application of cdl to image denoising, in: 2017 International Conference on Sampling Theory and Applications, IEEE, 2017, pp. 460–464.

[15] Y. Liu, X. Chen, R. K. Ward, Z. J. Wang, Image fusion with convolutional sparse representation, IEEE Signal Processing Letters 23 (12) (2016) 1882–1886.

[16] M. Li, Q. Xie, Q. Zhao, W. Wei, S. Gu, J. Tao, D. Meng, Video rain streak removal by multiscale convolutional sparse coding, in: Proceedings of the

IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6644–6653.

[17] K. Zhang, W. Zuo, L. Zhang, FFDNet: Toward a fast and flexible solution for CNN-based image denoising, IEEE Transactions on Image Processing 27 (9) (2018) 4608–4622.

[18] X. Jia, S. Liu, X. Feng, L. Zhang, Focnet: A fractional optimal control network for image denoising, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 6054–6063.

[19] Y. Zhang, K. Li, K. Li, B. Zhong, Y. Fu, Residual non-local attention networks for image restoration, arXiv preprint arXiv:1903.10082.

[20] D. Liu, B. Wen, Y. Fan, C. C. Loy, T. S. Huang, Non-local recurrent network for image restoration, Advances in Neural Information Processing Systems 31.

[21] M. Li, J. Liu, Y. Fu, Y. Zhang, D. Dou, Spectral enhanced rectangle transformer for hyperspectral image denoising, in: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 5805–5814.

[22] K. Gregor, Y. LeCun, Learning fast approximations of sparse coding, in: Proceedings of the 27th International Conference on International Conference on Machine Learning, 2010, pp. 399–406.

[23] J. R. Hershey, J. L. Roux, F. Weninger, Deep unfolding: Model-based inspiration of novel deep architectures, arXiv preprint arXiv:1409.2574.

[24] Z. Xu, J. Sun, Model-driven deep-learning, National Science Review 5 (1) (2018) 22–24.

[25] V. Monga, Y. Li, Y. C. Eldar, Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing, IEEE Signal Processing Magazine 38 (2) (2021) 18–44.

[26] Y. Yang, J. Sun, H. Li, Z. Xu, ADMM-CSNet: A deep learning approach for image compressive sensing, IEEE Transactions on Pattern Analysis and Machine Intelligence 42 (3) (2020) 521–538.

[27] M. Scetbon, M. Elad, P. Milanfar, Deep K-SVD denoising, IEEE Transactions on Image Processing 30 (2021) 5944–5955.

[28] H. Zheng, H. Yong, L. Zhang, Deep convolutional dictionary learning for image denoising, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 630–641.

[29] S. Tariyal, A. Majumdar, R. Singh, M. Vatsa, Deep dictionary learning, IEEE Access 4 (2016) 10096–10109.

[30] S. Mahdizadehaghdam, A. Panahi, H. Krim, L. Dai, Deep dictionary learning: A parametric network approach, IEEE Transactions on Image Processing 28 (10) (2019) 4790–4802.

[31] N. Janjušević, A. Khalilian-Gourtani, Y. Wang, CDLNet: Noise-adaptive convolutional dictionary learning network for blind denoising and demosaicing, IEEE Open Journal of Signal Processing 3 (2022) 196–211.

[32] D. Simon, M. Elad, Rethinking the CSC model for natural images, in: Advances in Neural Information Processing Systems, Vol. 32, 2019.

[33] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in: Proceedings of the 18th International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.

[34] C. Mou, Q. Wang, J. Zhang, Deep generalized unfolding networks for image restoration, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 17399–17410.

[35] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, M.-H. Yang, Restormer: Efficient transformer for high-resolution image restoration, in:

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 5728–5739.

[36] J. Mairal, F. Bach, J. Ponce, G. Sapiro, Online dictionary learning for sparse coding, in: Proceedings of the 26th Annual International Conference on Machine Learning, 2009, pp. 689–696.

[37] M. Elad, M. Aharon, Image denoising via sparse and redundant representations over learned dictionaries, IEEE Transactions on Image processing 15 (12) (2006) 3736–3745.

[38] X. Cao, J. Miao, Y. Xiao, Medical image segmentation of improved genetic algorithm research based on dictionary learning, World Journal of Engineering and Technology 05 (2017) 90–96.

[39] R. Yan, Y. Liu, Y. Liu, L. Wang, R. Zhao, Y. Bai, Z. Gui, Image denoising for low-dose CT via convolutional dictionary learning and neural network, IEEE Transactions on Computational Imaging 9 (2023) 83–93.

[40] T. Liu, A. Chaman, D. Belius, I. Dokmanić, Learning multiscale convolutional dictionaries for image reconstruction, IEEE Transactions on Computational Imaging 8 (2022) 425–437.

[41] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, Foundations and Trends® in Machine learning 3 (1) (2011) 1–122.

[42] Y. Luo, Y. Cai, J. Ling, Y. Ji, Y. Tie, S. Yao, Joint edge optimization deep unfolding network for accelerated MRI reconstruction, IEEE Transactions on Computational Imaging (2024) 1–14.

[43] L. Zhao, X. Wang, J. Zhang, A. Wang, H. Bai, Boundary-constrained interpretable image reconstruction network for deep compressive sensing, Knowledge-Based Systems 275 (2023) 110681.

[44] L. Zhao, J. Zhang, J. Zhang, H. Bai, A. Wang, Joint discontinuity-aware depth map super-resolution via dual-tasks driven unfolding network, IEEE Transactions on Instrumentation and Measurement 73 (2024) 1–14.

[45] W. Li, B. Chen, S. Liu, S. Zhao, B. Du, Y. Zhang, J. Zhang, D^3C^2-Net: Dual-domain deep convolutional coding network for compressive sensing, IEEE Transactions on Circuits and Systems for Video Technology 34 (10) (2024) 9341–9355.

[46] X. Wang, L. Zhao, J. Zhang, A. Wang, H. Bai, A wavelet-domain consistency-constrained compressive sensing framework based on memory-boosted guidance filtering, IEEE Transactions on Instrumentation and Measurement 73 (2024) 1–16.

[47] B. Tolooshams, A. Song, S. Temereanca, D. Ba, Convolutional dictionary learning based auto-encoders for natural exponential-family distributions, in: International Conference on Machine Learning, 2020, pp. 9493–9503.

[48] T. Meinhardt, M. Moller, C. Hazirbas, D. Cremers, Learning proximal operators: Using denoising networks for regularizing inverse imaging problems, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 1781–1790.

[49] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, R. Timofte, Plug-and-play image restoration with deep denoiser prior, IEEE Transactions on Pattern Analysis and Machine Intelligence 44 (10) (2021) 6360–6376.

[50] K. Zhang, W. Zuo, S. Gu, L. Zhang, Learning deep CNN denoiser prior for image restoration, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2017, pp. 3929–3938.

[51] M. Li, Y. Fu, J. Liu, Y. Zhang, Pixel adaptive deep unfolding transformer for hyperspectral image reconstruction, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 12959–12968.

[52] Y. Dong, D. Gao, T. Qiu, Y. Li, M. Yang, G. Shi, Residual degradation learning unfolding framework with mixing priors across spectral and spatial for compressive spectral imaging, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 22262–22271.

[53] Y. Tang, J. Li, L. Yue, X. Liu, Y. Li, Y. Xiao, Q. Yuan, A CNN-transformer embedded unfolding network for hyperspectral image super-resolution, IEEE Transactions on Geoscience and Remote Sensing 62 (2024) 1–16.

[54] T. Park, M.-Y. Liu, T.-C. Wang, J.-Y. Zhu, Semantic image synthesis with spatially-adaptive normalization, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2337–2346.

[55] Y. Wang, W. Yin, J. Zeng, Global convergence of ADMM in nonconvex nonsmooth optimization, Journal of Scientific Computing 78 (2019) 29–63.

[56] A. Beck, M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM Journal on Imaging Sciences 2 (1) (2009) 183–202.

[57] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, M.-H. Yang, L. Shao, Multi-stage progressive image restoration, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 14821–14831.

[58] S. Cheng, Y. Wang, H. Huang, D. Liu, H. Fan, S. Liu, Nbnet: Noise basis learning for image denoising with subspace projection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 4896–4906.

[59] 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 Eighth IEEE International Conference on Computer Vision, Vol. 2, 2001, pp. 416–423.

[60] E. Agustsson, R. Timofte, Ntire 2017 challenge on single image super-resolution: Dataset and study, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2017, pp. 126–135.

[61] K. Ma, Z. Duanmu, Q. Wu, Z. Wang, H. Yong, H. Li, L. Zhang, Waterloo exploration database: New challenges for image quality assessment models, IEEE Transactions on Image Processing 26 (2) (2016) 1004–1016.

[62] K. Zhang, W. Zuo, Y. Chen, D. Meng, L. Zhang, Beyond a gaussian denoiser: Residual learning of deep CNN for image denoising, IEEE Transactions on Image Processing 26 (7) (2017) 3142–3155.

[63] Z. Wang, A. Bovik, H. Sheikh, E. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Transactions on Image Processing 13 (4) (2004) 600–612.

[64] R. Franzen, Kodak lossless true color image suite, source: http://r0k.us/graphics/kodak 4 (2).

[65] L. Zhang, X. Wu, A. Buades, X. Li, Color demosaicking by local directional interpolation and nonlocal adaptive thresholding, Journal of Electronic Imaging 20 (2) (2011) 023016–023016.

[66] C. Tian, Y. Xu, W. Zuo, Image denoising using deep CNN with batch renormalization, Neural Networks 121 (2020) 461–473.

[67] Z. Xia, A. Chakrabarti, Identifying recurring patterns with deep neural networks for natural image denoising, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2020, pp. 2426–2434.

[68] H. Zhao, Y. Gou, B. Li, D. Peng, J. Lv, X. Peng, Comprehensive and delicate: An efficient transformer for image restoration, in: Proceedings

of the IEEE/CVF conference on computer vision and pattern recognition, 2023, pp. 14122–14132.

[69] J.-B. Huang, A. Singh, N. Ahuja, Single image super-resolution from transformed self-exemplars, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5197–5206.

[70] Q. Chen, Y. Wang, Z. Geng, Y. Wang, J. Yang, Z. Lin, Equilibrium image denoising with implicit differentiation, IEEE Transactions on Image Processing 32 (2023) 1868–1881.

[71] S. Nam, Y. Hwang, Y. Matsushita, S. J. Kim, A holistic approach to cross-channel image noise modeling and its application to image denoising, in: Proceedings of the Conference on Computer Vision and Pattern Recognition, 2016, pp. 1683–1691.