Wavelet Analysis Model Inspired Convolutional Neural Networks for Image Denoising

Ruotao Xu[a,c], Yong Xu[b,d,e], Xuhui Yang[b], Haoran Huang[b], Zhenghua Lei[b], Yuhui Quan[b,c,*]

[a]*Institute for Super Robotics, South China University of Technology, Guangzhou, China*
[b]*School of Computer Science and Engineering, South China University of Technology, Guangzhou, China*
[c]*Pazhou Laboratory, Guangzhou, China*
[d]*Peng Cheng Laboratory, Shenzhen, China*
[e]*Communication and Computer Network Laboratory of Guangdong, Guangzhou, China*

Abstract

In recent years, many image denoising methods have been proposed based on convolutional neural networks (CNNs). While these methods have shown continuous performance improvement by introducing various mechanisms and structures, their computational cost tends to become increasingly expensive, owing to the resulting complex network architectures. This paper aims at winning the trade-off between computational efficiency and denoising performance for CNN-based image denoisers. Towards this end, we draw inspirations from traditional variational models with wavelet analysis operators for CNN architecture design. A model-inspired CNN is proposed with four key modules: iterative encoding-decoding units inspired by the iterative denoising process, directional convolutions inspired by the separable wavelet filters, inception modules inspired by the multi-scale analysis of wavelets, and stage-wise connections inspired by the adding noise back operation. In experiments, our CNN shows high computational efficiency in both training and test, with competitive results to state-of-the-art approaches.

Keywords:
Image denoising, Wavelet analysis model, Deep learning, Neural networks

*Corresponding author
Email addresses: `xrt@scut.edu.cn` (Ruotao Xu), `yxu@scut.edu.cn` (Yong Xu), `csyoe@mail.scut.edu.cn` (Xuhui Yang), `csherry@mail.scut.edu.cn` (Haoran Huang), `cszhlei@mail.scut.edu.cn` (Zhenghua Lei), `csyhquan@scut.edu.cn` (Yuhui Quan)

1. Introduction

Image denoising is a fundamental problem in the field of image processing [1] and has practical values to many computer vision tasks [2–4]. It aims at removing the noise from a noisy image and meanwhile preserving image details such as edges and corners. Let $\boldsymbol{x}$ denote the ground-truth clean image and $\boldsymbol{y}$ the observed noisy image. Generally, noise corruption can be formulated as follows:

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n}, \tag{1}$$

where $\boldsymbol{n}$ denotes the measurement noise which is often assumed to be the additive Gaussian white noise (AWGN). Then, image denoising is about solving $\boldsymbol{x}$ from Eq. (1), given $\boldsymbol{y}$ as a known variable.

1.1. Motivations

Taking the advance of deep learning, many deep approaches (*e.g.* [5–7]) have been proposed for image denoising. These approaches use deep convolutional neural networks (CNNs) with various types of structures to learn the denoising process directly from a large set of noisy/clear image pairs. The design of the CNN structure is the key to such CNN-based approaches. Ideally, a practical CNN should have (a) fast execution time that satisfies the need of the real-time applications and (b) sufficient expressibility to learn the denoising mapping as well as good generalizability to unseen images.

While existing CNN-based approaches have shown continuous performance improvement by introducing various mechanisms and structures, their computational cost tends to become increasingly expensive owing to the resulting complex network architectures. This motivated us to develop a CNN that can win the trade-off between computational efficiency and denoising performance.

However, designing an effective yet efficient CNN is non-trivial and there is often no way to start. There are some approaches (*e.g.* [6, 8]) designing their CNNs for image restoration based on the iterative process of some traditional method. In this paper, we draw prior knowledge from the wavelet analysis model for image denoising, based on which a model-inspired CNN called SED-Net (Sequential Encoding-Decoding Network) is proposed. Benefiting from the prior knowledge from the well-established wavelet analysis model, SED-Net enjoys both computational efficiency and effectiveness simultaneously.

1.2. Basic Idea

Many existing approaches to image denoising (*e.g.* [9, 10]) estimate the clean $\boldsymbol{x}$ by solving variational models of a general form:

$$\min_{\boldsymbol{x}} \ \|\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \lambda\phi(\boldsymbol{W}\boldsymbol{x}), \tag{2}$$

where $\lambda > 0$ is a weight, $\boldsymbol{W}$ is a transform for generating some kind of image representation, and $\phi(\cdot)$ is a function for imposing the image priors derived from the statistics of natural images on the domain defined by $\boldsymbol{W}$. One popular choice for $\boldsymbol{W}$ is the wavelet tight frame [11] which has demonstrated its effectiveness and computational efficiency in many image restoration tasks, owing to its capability of multi-scale analysis and localization.

Let $\boldsymbol{W}$ be a wavelet tight frame satisfying $\boldsymbol{W}^\top\boldsymbol{W} = \boldsymbol{I}$, where $\boldsymbol{W}$ and $\boldsymbol{W}^\top$ are called the analysis and synthesis operators, respectively. Then the model in(2) becomes the so-called wavelet analysis model. We use the half-quadratic splitting [12] to solve the model in (2). By introducing an auxiliary variable $\boldsymbol{h}$, the solver iteratively calculates

$$\begin{cases} \boldsymbol{h}^{(t+1)} = \boldsymbol{W}^\top\Psi(\boldsymbol{W}\boldsymbol{x}^{(t)}) \\ \boldsymbol{x}^{(t+1)} = \omega^{(t)}\boldsymbol{h}^{(t+1)} + (1-\omega^{(t)})\boldsymbol{y} \end{cases} \tag{3}$$

for $t = 0, 1, \cdots$, where Ψ is an operator associated with ϕ and $\{0 < \omega^{(t)} < 1\}_t$ is a parameter sequence. For instance, Ψ is a scaling function when ϕ is the ℓ_2 norm, a soft thresholding function when ϕ is the ℓ_1 norm, and a hard thresholding function when ϕ is the ℓ_0 norm; see also Sec. 3.2 for more details. The denoising process defined by (3) iteratively alternates two sub-processes: (a) update of $\boldsymbol{h}$: manipulation in the wavelet domain defined by the wavelet transform $\boldsymbol{W}$ and then turn the manipulated wavelet coefficients back to the image domain by the inverse wavelet transform $\boldsymbol{W}^{-1} = \boldsymbol{W}^\top$; (b) update of $\boldsymbol{x}$: merging $\boldsymbol{h}^{(t+1)}$ with the input image $\boldsymbol{y}$ using weights $(\omega^{(t)}, 1-\omega^{(t)})$. There are three key ingredients in such a process: (a) iterative processing; (b) processing in the wavelet domain (*i.e.* $\Psi(\boldsymbol{W}\boldsymbol{x}^{(t)})$) and transforming back (*i.e.* $\boldsymbol{W}^\top\boldsymbol{z}^{(t+1)}$); and (c) merging with input noisy image, also often called adding the noise back, *i.e.*, $\omega\boldsymbol{h}^{(t+1)} + (1-\omega)\boldsymbol{y}$.

Inspired by the key ingredients of the process defined in (3) as well as the construction of the involved 2D wavelet transform, we construct our SED-Net with the following modules:

- *Sequential encoding-decoding units.* In the update of $\boldsymbol{h}$, the estimated image $\boldsymbol{x}$ is first transformed to wavelet domain by the analysis operator $\boldsymbol{W}$, manipulated with $\Psi(\cdot)$, and then transformed back into image domain by the corresponding synthesis operator $\boldsymbol{W}^\top$. By viewing the wavelet transform and inverse wavelet transform as an encoder and decoder respectively, we define the backbone of SED-Net by a series of encoding-decoding units to simulate this iterative wavelet-based processing.

- *Directional convolutions.* The 2D wavelet filters for $\boldsymbol{W}$ are often constructed by the tensor product of two 1D wavelet filters (horizontal/vertical in the 2D plane). This enables generating a 2D convolution kernel of size $M \times N$ using parameters whose number is less than MN. However, since oriented kernels except the horizontal/vertical ones are non-separable kernels, the 2D kernels constructed from the 1D ones cannot have rich orientations. In other words, employing only 1D kernels in a CNN to simulate 2D convolutions cannot effectively extract the rich-oriented structures in images. Thus, we combine 1D convolutions and 2D ones in the convolutional layers of SED-Net.

- *Inception-like module.* Multi-scale analysis is known to be the most important property of wavelets [13], which enables the extraction of image structures from rough scales to fine scales. Many studies have shown that the multi-scale analysis benefits for denoising images with varying-scale structures [13]. Inspired by the multi-scale analysis capability of wavelets, we introduce the inception-like structure into each encoding-decoding unit, which contains convolutions with various kernel sizes.

- *Stage-wise connections.* The operation of adding noise back, as seen in Eq (3), is commonly utilized in the numerical solvers of variational denoising models. Typically, denoising involves low-pass filtering, which results in the loss of certain image details. To counteract this, the addition of the noisy image to the denoised intermediates can be employed to reintroduce image details and lead to final results with finer details. In light of this, we add a skip connection between the noisy image and the output of the final encoder-decoder (ED) stage. Simulating adding noise back operation, this connection preserves details by transmitting details from the noisy image to the output of the final ED stage. Additionally, due to the data-driven property of neural networks, the ED units hence may also produce desired details for recovery. Therefore, we also introduce skip connections between the output of each ED and the final one. These connections form the proposed stage-wise connections.

To summarize, existing CNNs cannot balance effectiveness and computational efficiency well. This work investigates a CNN-based denoiser that can enjoy both state-of-the-art performance and low computational cost. Different from the existing ones, the proposed CNN is inspired by variational models. Such a model-inspired design benefits much performance improvement. These benefits are demonstrated on standard benchmark datasets, and the experimental results have shown the solid performance of our proposed CNN.

2. Related Work

Traditional methods exploit certain image priors to overcome the undeterminedness of the denoising problem. Many methods use the sparsity prior which assumes sparse intensity changes of images, and implement the prior by minimizing the ℓ_1 norm of image coefficients in some domains; see *e.g.* total variation-based methods [14–17] and wavelet-based methods [18]. Sparse representation methods [19, 20] suppose clean image patches can be sparsified under a certain dictionary and learn the dictionary with sparse constraints. Another popular image prior is the nonlocal self-similarity (NSS), utilizing recurrent patches within spatial neighborhood [21, 22], across scale space [23], on external images [24]. There are some approaches learning the parameterized distribution of image patches on clear images and using the learned distribution as an image prior; see *e.g.* [25].

In recent years, supervised learning has emerged as a promising approach to image denoising. Schmidt *et al.* [26] unrolled the traditional models into a learnable cascade denoising process and learned such a process from training data. Chen *et al.* [27] unrolled the diffusion process into a learnable one, with improvement over Schmidt *et al.*'s method. These two methods are closely related to ours as they also draw inspirations from traditional models. Leveraging deep learning, our approach shows superior performance over them in the experiments.

The deep-learning-based methods for image denoising have shown outstanding performance over learning-based methods. The early work can be traced back to Burger *et al.* [28] that learns a multi-layer perceptron to map noisy patches onto clean ones. The most representative approach is Zhang *et al.*'s DnCNN [29], which is a simple yet practical CNN for image denoising and has been used as a benchmark in many studies. To improve efficiency and handle spatially-varying noises, the FFDNet [30] takes downsampled subimages and a tunable noise level map as input.

There are some CNN-based approaches exploiting the NSS prior. Yang and Sun [8] unrolled the BM3D [8] method into a CNN. Ploetz *et al.* [31] introduced a neural nearest neighboring block. While introducing the NSS prior to CNNs brings performance gain, the resulting CNN is less efficient compared to DnCNN and has heavy requirements on computational resources (*e.g.* memory).

Most existing works focus on improving network architectures. Tai *et al.* [32] introduced a persistent memory network (MemNet) and exploited dense connection for better preserving the high-frequency information during denoising. Yu *et al.*[33] proposed a deep iterative down-up network. Park *et al.* [34] studied a densely connected hierarchical network by using a modified U-Net architecture. Fang *et al.* [35] proposed a multilevel edge features guided CNN. Quan *et al.* [36] proposed a complex-valued CNN to exploit the merits of complex-valued operations, such as the compactness of convolution given by the tensor product of 1D complex-valued filters and the nonlinear activation on phases. Li *et al.* [37] explored the degradation mechanism of the noisy image and proposed a lightweight network to progressively remove noise.

There are some studies drawing inspirations from wavelet to design CNNs for image denoising. Inspired by the lifting wavelet transform, Huang *et al.* [38] proposed a lifting-based invertible CNN, which learns a non-linear redundant transform with perfect reconstruction property to reduce noise. Liu *et al.* [39] presented a multi-level wavelet CNN for a better trade-off between the receptive field size and the computational efficiency. To further incorporate the power of wavelet transform, Tian *et al.* [40] introduced a series of cascading wavelet transforms and enhancement blocks, enhancing the features in the wavelet domain. In comparison, our work goes beyond the wavelet transform, but further looks at a whole wavelet analysis model for denoising, which can win the trade-off between computational efficiency and denoising performance.

3. Preliminaries

Throughout this paper, bold upper letters are used for matrices, bold lower letters for column vectors, and calligraphic letters for operators. The notations $\mathbf{0}$ and $\boldsymbol{I}$ denote the zero matrix and the identity matrix with appropriate sizes respectively. The ℓ_p norm of a vector is denoted by $\|\cdot\|_p$. Given a sequence $\{\boldsymbol{v}^{(t)}\}_{t\in\mathbb{N}}$, $\boldsymbol{v}^{(t_0)}$ denote the t_0-th element in the sequence.

3.1. Wavelet Tight Frames

Since framelets are considered in the proposed method, we first briefly present some basics of framelets for image processing. Interested readers are referred to [41, 42] for more details. Let $\mathbb{H}$ be a Hilbert space with the usual inner product

$\langle\cdot,\cdot\rangle$ and norm $\|\cdot\|$. A sequence $\{\phi_n\}_{n\in\mathbb{Z}}\subset\mathbb{H}$ is a frame for $\mathbb{H}$ if there exist two positive constants a and b such that

$$a\|f\|_2^2\leq\sum_{n\in\mathbb{Z}}|\langle\phi_n,f\rangle|^2\leq b\|f\|_2^2,\quad\forall\, f\in\mathbb{H}. \tag{4}$$

A frame $\{\phi_n\}_{n\in\mathbb{Z}}$ is called a tight frame for $\mathbb{H}$ when $a=b$. Frame can be viewed as a generalization of Riesz Basis and tight frame is a redundant system that generalizes orthogonal basis.

There are two operators associated with a given frame $\{\phi_n\}_{n\in\mathbb{Z}}$: the analysis operator $\mathcal{W}$ defined by

$$\mathcal{W}: f\in\mathbb{H}\longrightarrow\{\langle f,\phi_n\rangle\}\in\ell^2(\mathbb{Z}), \tag{5}$$

and its adjoint operator $\mathcal{W}^*$, also called the synthesis operator, defined by

$$\mathcal{W}^*:\{c_n\}\in\ell^2(\mathbb{Z})\longrightarrow\sum_n c_n\phi_n\in\mathbb{H}. \tag{6}$$

Wavelet tight frames are arguably the most frequently-used frames in image processing. A wavelet tight frame is a tight frame system generated by the shifts and dilations of a finite set of generators, which can be implemented by convolutions with a set of filters that has some properties. Given a filter $\boldsymbol{a}\in\ell_2(\mathbb{Z})$, define the linear convolution operator $\mathcal{S}_{\boldsymbol{a}}:\ell_2(\mathbb{Z})\to\ell_2(\mathbb{Z})$ by

$$[\mathcal{S}_{\boldsymbol{a}}\boldsymbol{v}](n)=[\boldsymbol{a}*\boldsymbol{v}](n)=\sum_{k\in\mathbb{Z}}\boldsymbol{a}(n-k)\boldsymbol{v}(k),\forall\boldsymbol{v}\in\ell_2(\mathbb{Z}). \tag{7}$$

For a set of wavelet filters $\{\boldsymbol{a}_i\}_{i=1}^m\subset\ell_2(\mathbb{Z})$, the corresponding analysis operator has the matrix representation as follows:

$$\boldsymbol{W}=[\mathcal{S}_{\boldsymbol{a}_1(-\cdot)}^\top,\mathcal{S}_{\boldsymbol{a}_2(-\cdot)}^\top,\ldots,\mathcal{S}_{\boldsymbol{a}_m(-\cdot)}^\top]^\top, \tag{8}$$

and by definition, the corresponding synthesis operator is $\boldsymbol{W}^\top$.

The wavelet frames formed by multi-resolution analysis are often referred to as framelet systems, and the associated filters are called framelet filters. Framelet filters can be effectively constructed by the Unitary Extension Principle (UEP) [43]. For instance, the 1D single-level linear B-spline wavelet tight frame constructed by UEP has the following three filters:

$$\boldsymbol{a}_1=\frac{1}{4}(1,2,1)^\top;\ \boldsymbol{a}_2=\frac{\sqrt{2}}{4}(1,0,-1)^\top;\ \boldsymbol{a}_3=\frac{1}{4}(-1,2,-1)^\top. \tag{9}$$

In the 2D case, the framelet filters can be constructed via the tensor product of the 1D versions. In this paper, we consider the separable property and the multi-scale design of 2D framelets, and propose the directional convolution and inception-like module under its inspiration.

3.2. Wavelet Analysis Model and Its Numerical Solver

Since unknowns are much more than the equations in the denoising problem (1), effective priors on the clean image $\boldsymbol{x}$ are needed to solve the system. Most existing regularization methods can be classified into two categories. One is the so-called synthesis-based approach [44, 45], whose model usually has the following general form:

$$\min_{\boldsymbol{x}} \ \|\boldsymbol{x}-\boldsymbol{y}\|_2^2+\lambda\phi(\boldsymbol{c}) \qquad s.t.\ \boldsymbol{x}=\boldsymbol{W}^\top\boldsymbol{c}, \tag{10}$$

Recall that $\boldsymbol{y}$ denotes the noisy observation. The synthesis-based model assumes that the clean image can be synthesized by some specific dictionary $\boldsymbol{W}^\top$ and the regularization is imposed by the coefficients $\boldsymbol{c}$. The other one is the so-called analysis-based approach [14, 18], which can be written as

$$\min_{\boldsymbol{x}} \ \|\boldsymbol{x}-\boldsymbol{y}\|_2^2+\lambda\phi(\boldsymbol{W}\boldsymbol{x}). \tag{11}$$

Different from the synthesis-based model, the analysis-based model assumes the clean image has some specific properties(*e.g.*, sparsity), under the carefully-designed transform $\boldsymbol{W}$. It is empirically observed that for image restoration, the estimated images obtained by the synthesis-based model usually have some visually unpleasant artifacts. In comparison, the analysis-based approach usually has fewer artifacts as they directly ensure the properties of the estimated images. Interested readers are referred to [11, 46] for more details on the two types of regularization models and their connections.

In this paper, we look at the analysis-based model and consider wavelet tight frame as the transform $\boldsymbol{W}$. The resulting model (11) is the so-called wavelet analysis model. We use the half-quadratic splitting [12] as the numerical solver. By introducing an auxiliary variable $\boldsymbol{z}$, the problem (11) can be reformulated as

$$\min_{\boldsymbol{x},\boldsymbol{z}} \ \|\boldsymbol{x}-\boldsymbol{y}\|_2^2+\beta\,\|\boldsymbol{z}-\boldsymbol{W}\boldsymbol{x}\|_2^2+\lambda\phi(\boldsymbol{z}), \tag{12}$$

which is equivalent to (11), when $\beta\to\infty$. Therefore, to solve the problem (11), we can iteratively solve the $\boldsymbol{z}$-subproblem and $\boldsymbol{x}$-subproblem in (12) with increas-

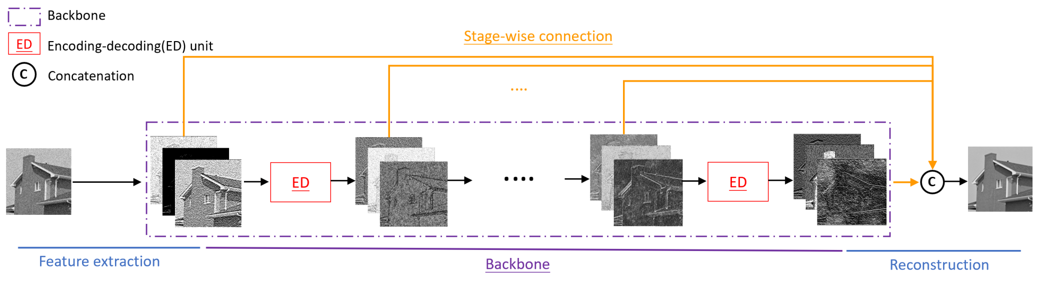


Figure 1: Illustration of the architecture of proposed SED-Net.

ing β as follows:

$$\begin{cases} \boldsymbol{z}^{(t+1)} = \arg\min_{\boldsymbol{z}} \beta^{(t)} \left\|\boldsymbol{z} - \boldsymbol{W}\boldsymbol{x}^{(t)}\right\|_2^2 + \lambda\phi(\boldsymbol{z}) \\ \boldsymbol{x}^{(t+1)} = \arg\min_{\boldsymbol{x}} \ \|\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \beta^{(t)} \left\|\boldsymbol{z}^{(t+1)} - \boldsymbol{W}\boldsymbol{x}\right\|_2^2 \end{cases}, \tag{13}$$

where $\{\beta^{(t)} > 0\}_t$ is an increasing sequence. Both sub-problems in (13) have explicit solutions, which can be written as

$$\begin{cases} \boldsymbol{z}^{(t+1)} = \Psi(\boldsymbol{W}\boldsymbol{x}^{(t)}) \\ \boldsymbol{x}^{(t+1)} = \omega^{(t)}\boldsymbol{W}^\top\boldsymbol{z}^{(t+1)} + (1-\omega^{(t)})\boldsymbol{y} \end{cases}, \tag{14}$$

where $\Psi(\cdot)$ is dependent on the regularization function $\phi(\cdot)$ and the weight $\lambda/\beta(t)$ and $\omega^{(t)} = \frac{\beta^{(t)}}{\beta^{(t)}+1}$. Note that the solver (14) can be rewritten into (3) by substituting $\boldsymbol{z}$ with $\boldsymbol{h} = \boldsymbol{W}^\top\boldsymbol{z}$. In this paper, we consider two key facts in the iteration (14): i) the analysis operator $\boldsymbol{W}$ and the synthesis operator $\boldsymbol{W}^\top$ are iteratively performed. ii) the noise is added back to the estimation in each iteration. Inspired by these facts, we develop the sequential encoder and decoder backbone and the stage-wise connection in the proposed denoising network.

4. Proposed Method

The architecture of the proposed SED-Net for image denoising is illustrated in Fig. 1. The network comprises three components: 1) a feature extraction layer, which maps the image into the feature domain, 2) a backbone, implemented as a sequential concatenation of encoding-decoding units inspired by solver (3), 3) a reconstruction layer, which maps the features back into the image domain. Below, we first detail the proposed ED unit, followed by a description of the network framework.

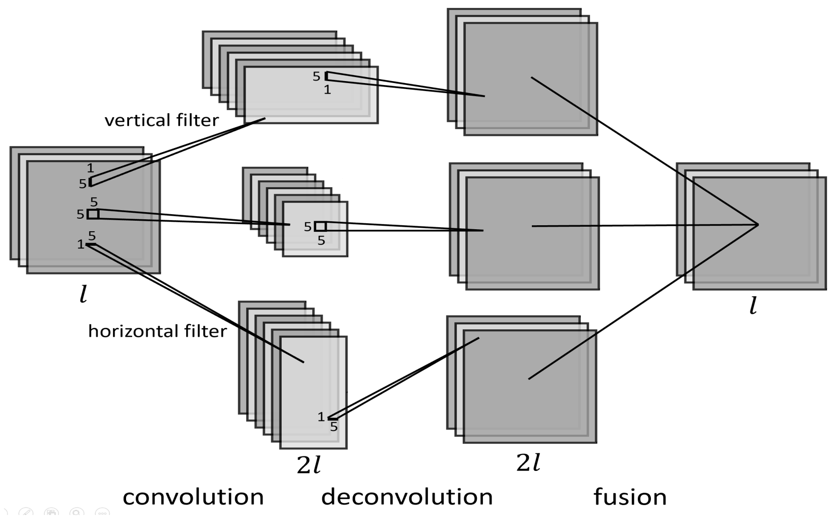


Figure 2: Illustration of directional convolution module.

4.1. Encoding-Decoding Unit

Consider an iteration of (3), where $\boldsymbol{W}$ and $\boldsymbol{W}^\top$ can be interpreted as an encoding process and a decoding process respectively. We thus implemented an encoding-decoding unit, where we sequentially connect a strided convolutional layer (simulating $\boldsymbol{W}$), Rectified Linear Unit (ReLU, simulating Ψ), and a deconvolutional layer with a fusion mechanism (simulating $\boldsymbol{W}^\top$). Rather than the normal convolutional and deconvolutional layers, we introduce the directional convolutions and inception modules under the inspiration of the solver (3), whose details are provided as follows.

Directional Convolutions The wavelet filters are generated by the tensor product of 1D ones. Compared to the 2D ones, using 1D filters can use fewer parameters to generate a 2D filter with a larger receptive field. Besides, images often have horizontal or vertical edges. Using 1D filters can better handle or preserve the horizontal/vertical image edges. However, purely using 1D filters can only generate filters with limited directions (horizontal or vertical). To enrich the orientations of filters, normal convolutions are needed.

Inspired by the above discussion, we embed directional convolutions into the ED unit, as shown in Fig. 2. Three types of convolutions, including horizontal convolution, vertical convolution, and normal convolution, are performed simultaneously at the encoding stage with a stride equal to two along x-axis, y-axis, or both. In the decoding stage, we apply the corresponding deconvolution to the three

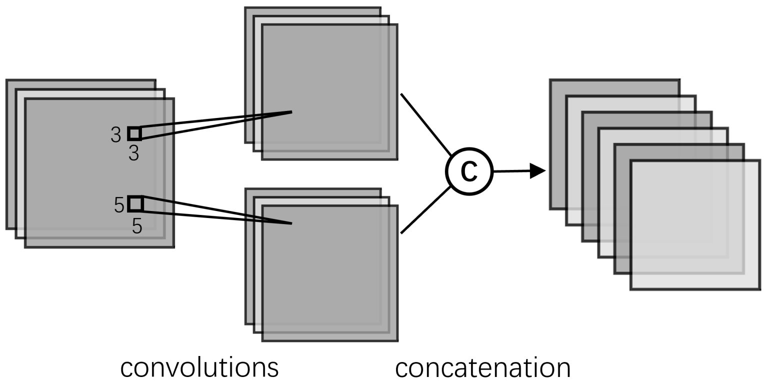


Figure 3: Simplified inception module. The feature maps obtained by convolutions with kernels of various sizes are concatenated before being transferred to the next layer.

intermediate feature maps to keep outputs the same size. In the implementation, the depth of the intermediate feature maps is all 128 while the decoding results have a depth of 64. The cascade of combined directional convolution and normal convolution can model both long horizontal/vertical edges and edges in other directions and corners. We experimentally found that embedding the directional convolution component to the backbone substantially increases the performance.

The aforementioned fusion layer is applied to merge the three feature maps from the decoding block. For simplicity, we define the fusion layer as the summation over channels, without learnable weights in the summation.

Inception Module One characteristic of wavelet is its multi-scale analysis capability, which can adapt to image patterns in different scales, *e.g.* larger convolution kernel is suitable for flat regions but not for detailed textures. Inspired by such a characteristic, we employ the simplified inception module to simulate the multi-scale analysis in each ED unit. We split each convolution/deconvolution into a convolution with a small kernel and a convolution with a large kernel, *e.g.* squared kernels in size of 3×3 and 5×5. See Fig. 3 for the structure. The same operation is also applied to directional convolution/deconvolution, *e.g.* vertical kernels in size of 3×1 and 5×1. The decoding feature maps from different types of kernels are also fused by summation. Assume the depth of the feature maps without using the inception module is $2l$, then the depth of each split feature map after using the inception module is l. The repeatedly mixed convolutions can form receptive fields of various sizes.

Unit Design In SED-Net, both the proposed directional convolutions and in-

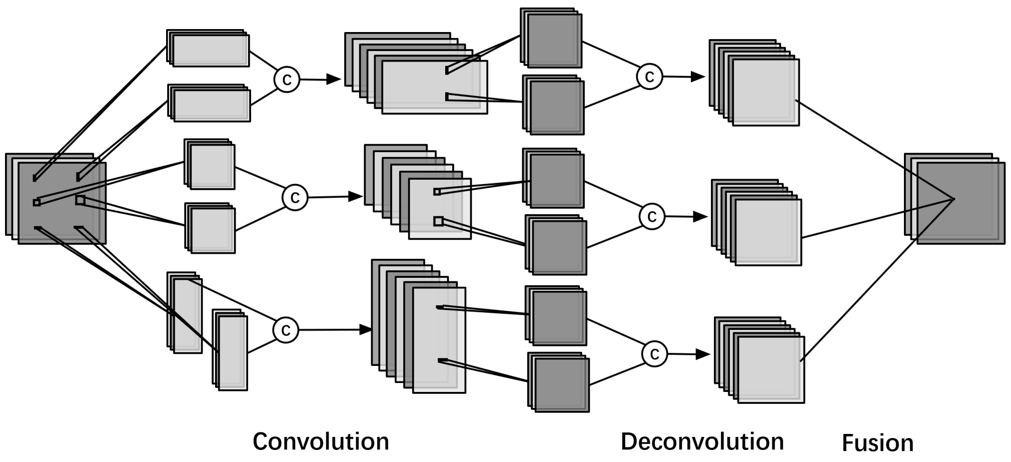


Figure 4: Illustration of the proposed ED Unit, which combines the directional convolutions and inception modules.

ception module are introduced into the convolutional and deconvolutional layers. For each unit, there are three types of convolutions: horizontal convolution, vertical convolution, and normal convolution. Each type of convolution can be split into two independent convolutions with the small or large kernel. Take horizontal convolution for example, in the encoding stage, two convolutions with kernel size 1×3 and 1×5 are performed respectively, both with stride setting as $(1, 2)$. Since the size of the input feature is $m \times n \times 64$, so the two encoding features are of size $m \times (n/2) \times 64$, which will be concatenated along the depth dimension to form an encoding feature of size $m \times (n/2) \times 128$. Then the horizontal deconvolutions adopt the same setting and finally produce a decoding feature of size $m \times n \times 64$, which is the same size as the input feature. Similarly, we can obtain the other two decoding features with different kernel sizes and strides. Finally, the three types of decoding results of size $m \times n \times 64$ are fused by element summation. In particular, the width and height of the feature map after encoding are shrunk by half while the number of feature channels is doubled to keep more information. See Fig. 4 for an illustration.

4.2. Network Architecture

Sequential ED Units Considering the iterative process of (3), we implemented SED-Net as a sequential concatenation of ED units, as shown in Fig. 5. Recall that the input and output of an encoding-decoding unit are of the same size, so the number of built-in encoding-decoding units can be arbitrary.

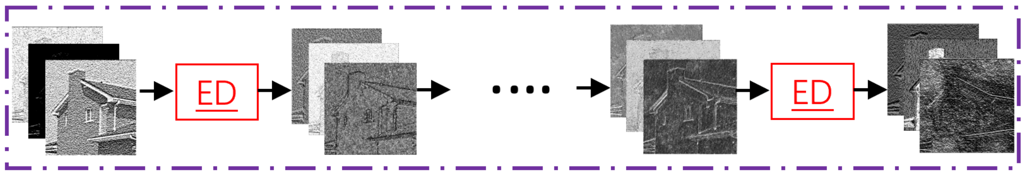


Figure 5: Illustration of the sequential ED units.

Stage-wise Connection The operation of adding the noise back is critical to the process of (3). Therefore, we add skip connections to the CNN. As shown in Fig. 1, the feature of the noisy image (output of the feature extraction layer) is combined with decoding features of ED units to generate the clean image. This connection way is not strictly the same as (3). For convenience, we flexibly merge these features in the last layer so that we do not have to merge ”noise” with decoding results everytime. The training process is more stable and the denoising result is better after changing the connection way. We named this connection way as stage-wise connection since each ED unit is like a simple denoising stage.

There are several benefits using stage-wise connections. First, skip connection naturally helps alleviate the gradient vanishing or exploding problems. Second, the decoding results with different depths have complementary information about image details. By combining information from layers in different depths, the stage-wise connection can help preserve the image details during the denoising process.

Architectural Details To preserve the spatial size of feature maps, we use zero padding for the neighborhood outside input feature maps before convolution. In the feature extraction module, a convolutional layer with a kernel size of 3×3 and 64 output channels followed by a rectified linear unit (ReLU) activation function is performed on the input images to extract image features. The backbone contains 9 encoding-decoding units, which brings 18 convolutional layers. Batch normalization and ReLU activation functions are performed after each convolutional layer of the encoding-decoding unit. Subsequently, the reconstruction module is employed, which consists of a convolutional layer with a kernel size of 3×3 without an activation function, yielding one output channel for a gray image. Therefore, the proposed SED-Net has 20 convolutional layers in total.

5. Experiments

5.1. Experiment Setting

Datasets. We use the same training dataset as TNRD [27] but without cropping the original images. The raw dataset contains 400 gray images of size 481×321 or 321×481. For each image, we crop 126 small images of size 64×64 by setting the cropping stride as 30 pixels. These cropped images are then augmented in two ways, flipping along the y-axis and rotating by four angles ($0°$, $90°$, $180°$ and $270°$). Therefore, one small image can generate 8 samples. 403200 training samples are obtained after augmentation. Following DnCNN, two popular testing datasets Set12 [21] and BSD68 [47] are utilized to evaluate the performance. Note that the images of BSD68 are not included in the training dataset although raw images are also from the Berkeley segmentation dataset.

Implementation Details. As well as most CNNs for image denoising, Adam optimizer [48] is chosen to minimize the mean squared error in this paper. We train 60 epochs for the SED-Net with a mini-batch size of 128. Piece-wise learning rate is adopted for training, which is 10^{-3} for the first 30 epochs, 10^{-4} for the next 20 epochs, and 10^{-5} for any remaining epochs. There are 3150 batches in total, but only 2500 batches are sampled for training per epoch. All filters in convolutional/deconvolutional layers are randomly initialized.

Hardware Configuration. All the experiments are conducted on a PC with an i7-7700K CPU and a Titan V GPU.

5.2. Ablation Study

To verify the effectiveness of the sequential ED units, we construct a baseline backbone using ED units with normal convolutions. Denoising networks using the baseline backbone and a plain CNN are trained for comparison. Both of them have a feature extraction layer and a reconstruction layer. Apart from this, the baseline has 5 iterative encoding-decoding units and the plain CNN has 10 convolutional layers in the middle to ensure that the two CNNs have the same depth.

The number of output channels of each convolution is set to 64 except for the encoding convolutional layer and the reconstruction layer. The kernel size is 5×5. Other proposed components are excluded to eliminate interference. The number of training epochs is set to 40 and the mini-batch size is set to 64 since fewer parameters need to be optimized. The settings of batch normalization, activation function, and other unmentioned parameters are consistent with those mentioned in Section 4.2 and section 5.1.

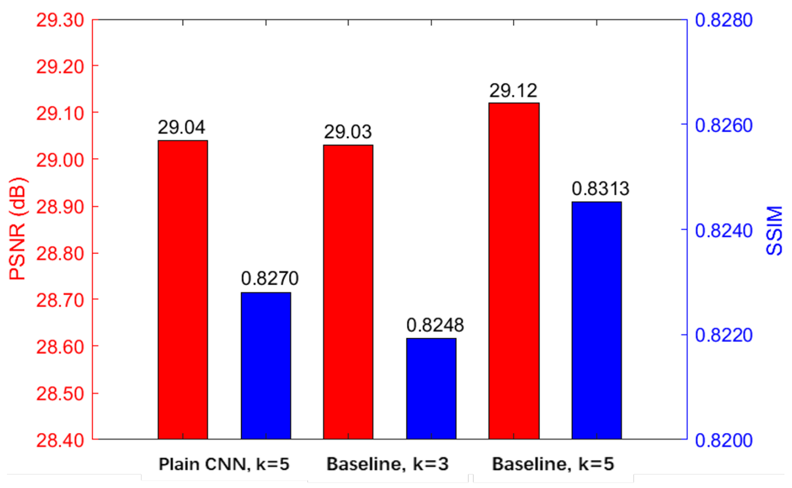


Figure 6: Average PSNR and SSIM of the plain CNN and the baseline over BSD68. $\sigma = 25$. k is kernel size.

Experiment results are shown in Fig. 6. With a smaller kernel size 3, the baseline is comparable to the plain CNN in terms of PSNR but has a lower SSIM value. A bigger kernel size is of benefit to capture the structural information. However, the baseline achieves a significant gain of about 0.08dB by PSNR and 0.0043 by SSIM on BSD68 if the same kernel size 5 is used. It might be because the encoding process is trained to better ignore the ”noise” signal and meanwhile learn a compact representation of the clean signal so as to remove noise and protect structures at the same time. Besides, the computational complexity of the baseline backbone is half that of the plain CNN with the same depth and same kernel size thanks to the convolution stride at encoding stages. The comparison result demonstrates the encoding-decoding unit's potential for image denoising.

Next, we conduct several experiments to study other proposed components' influence on performance. The following three components are selectively applied to the baseline. The resulting networks follow the same setting as the baseline if not particularly mentioned. Backbones with different configurations are trained and then evaluated on BSD68. Experiment results are shown in Table 1.

Directioinal Convolution (DC). To prevent edges or textures from being destroyed by ”averaging”, directional convolution/deconvolution is added to enhance details preservation. The original convolutions in the encoding-decoding unit of the baseline are replaced by mixtures of two types of directional convolutions and one normal convolution, as shown in Fig. 2.

Table 1: Influence analysis on proposed components. Average PSNR/SSIM on BSD68 with noise level as 25, the number of parameters and GFLOPs are reported. The symbol ✓ means embedding the corresponding component into the baseline backbone.

Setting	DC	IM	SC	PSNR	SSIM	#Params(M)	GFLOPs
Baseline				29.12	0.8313	3.69	151.15
Use DC	✓			29.16	0.8333	5.16	223.81
Use IM		✓		29.10	0.8309	2.51	102.83
Use SC			✓	29.22	0.8343	3.69	151.45
Not SC	✓	✓		29.16	0.8318	3.69	160.96
Not IM	✓		✓	29.33	0.8366	5.17	224.11
Not DC		✓	✓	29.29	0.8354	2.51	103.13
SED-Net	✓	✓	✓	29.32	0.8359	3.69	161.26

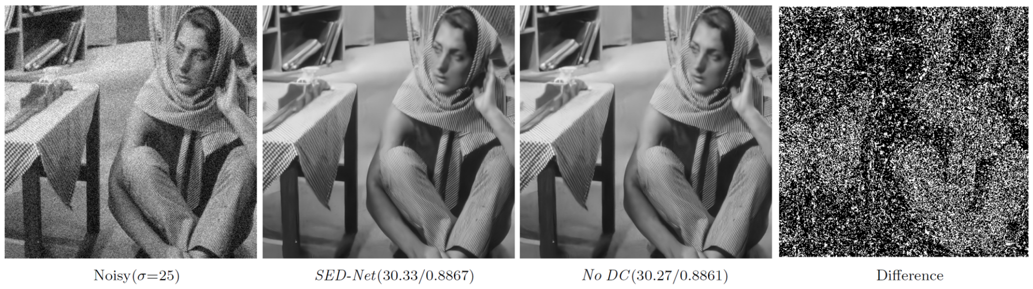


Figure 7: DC's influence on prediction.

The results verify the positive role of DC in denoising. As we can see, adding DC into the baseline leads to good improvements over both PSNR and SSIM, removing DC from SED-Net decreases 0.03dB by PSNR and 0.002 by SSIM. It is not surprising that the changed value is relatively small because fine-grained edges and textures have a limited influence over PSNR and SSIM. And the DC's mission is to recover clear edges and textures. We further visualize the denoising results in Fig. 7 to see DC's practical impact. There are four subfigures, including the noisy Barbara image, its two predictions from *SED-Net* and *Not DC* respectively, and a difference image. White pixels in the difference image indicate the corresponding position where *SED-Net* predicts a better pixel value than *Not DC*. It is obvious that clothes with fine textures are better recovered by *SED-Net*.

Inception Module (IM). The inception module is involved in the baseline to reduce the number of parameters. We quantify the resulting effect by changing original convolutions into inception modules. The simplified inception module contains a small kernel (3×3) and a large kernel (5×5), both with half output channels compared to the corresponding convolutional layer. Thus, The proposed IM has an intermediate number of parameters between the convolutions with small kernel size and large kernel size.

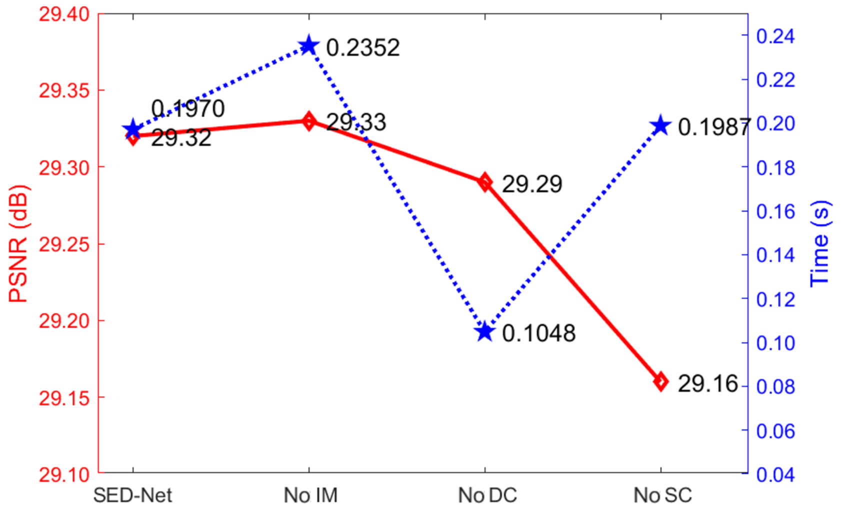


Figure 8: Different models' average running time on BSD68. The red curve represents PSNR values and the blue curve represents the average running time on BSD68.

Compared to the baseline with large kernel size, using IM not only reduces a great number of parameters but also achieves competitive performance. The PSNR's decrease is small (29.12dB versus 29.10dB) while SSIM decreases by 0.0004 (0.8313 versus 0.8309). Unlike these two cases, the baseline with small kernel size 3 has poor performance (29.03dB PSNR and 0.8248 of SSIM), which can be seen in Fig. 6. As the 5th column of Table 1 shows, the SED-Net has an average PSNR of 29.32dB. If removing IM from SED-Net, the average PSNR is 29.33dB. In summary, in our network, using IM will only cause slight PSNR and SSIM decrease. Fig. 8 presents a comparison of different models to evaluate the influence of the proposed components on both runtime efficiency and reconstruction quality. Among the four compared models, *SED-Net* and *Not IM* have the highest PSNR. But using IM can cut down about 30% parameters and thus accelerate denoising time. So we decide to embed IM into SED-Net as a trade-off between good denoising performance and computational efficiency.

Stage-wise Connection (SC). Extracted features of the first layer and features from other decoding layers are concatenated before being inputted into the last reconstruction layer. These connections are removed if SC is not ✓ at the table. Only the last decoding feature will be sent to the reconstruction layer.

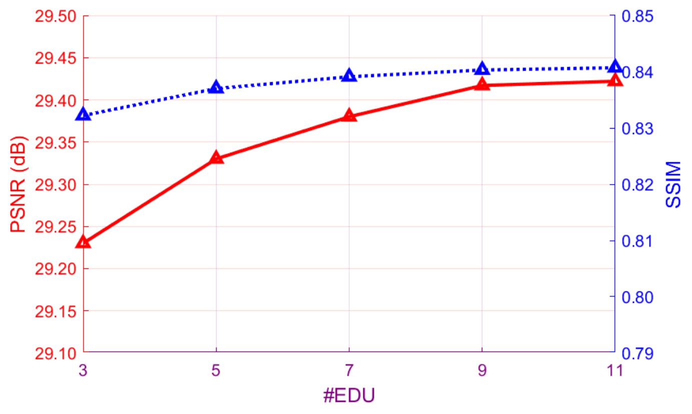


Figure 9: Average PSNR and SSIM on BSD68. $\sigma = 25$. #EDU means the number of Encoding-decoding units. The red curve represents PSNR values and the blue curve represents SSIM index.

Remarkably, it appeared that the SC component is of great importance for SED-Net, because removing SC from SED-Net causes large decrease in both PSNR and SSIM. The result of *Use SC* and *Baseline* also illustrates the SC's great impact on performance. The reason might be that the gradients are hard to be propagated to the shallower layers in training and the images' structures are inevitably being destroyed as the network goes deeper without short connections. By sparsely connecting features of various depths, the image structure's feature of shallow layers can be well preserved which is complementary to that of deep layers.

5.3. Comparison with SOTA

It is widely accepted that a deeper network has stronger learning ability. Recall that the iterations of encoding-decoding units in the SED-Net are flexible. So we first conducted an experiment to seek out an appropriate depth for the SED-Net. Fig. 9 shows experimental results in relation to the number of encoding-decoding units. Experiment results indicate that 9 is an appropriate and acceptable number for consideration of model size and performance.

Then we compared the proposed method with some state-of-the-art approaches. Twelve well-known and representative denoising methods, including the old but famous engineered method BM3D [21], the weighted nuclear norm minimization method (WNNM [49]), the classical learning-based method TNRD [27], and nine outstanding CNNs: DnCNN [29], MemNet [32] and NLRN [52], FFDNet [30], NLED [50], DRUNet [51], CDNet [36],DRUNet [51], WINNet [38] and MWD-CNN [40] are picked. Among them, WINNet and MWDCNN are two recently-

Table 2: Average PSNR/SSIM results of different Gaussian denoising methods. The best two results are highlighted in red and blue respectively.

Dataset	Method	$\sigma = 15$	$\sigma = 25$	$\sigma = 50$
Set12	BM3D [21]	32.37/0.8952	29.97/0.8504	26.72/0.7676
	WNNM [49]	32.70/0.8952	30.28/0.8557	27.05/0.7775
	TNRD [27]	32.50/0.8958	30.06/0.8512	26.81/0.7680
	NLED [50]	32.61/.08983	30.20/0.8566	26.91/0.7702
	DnCNN [29]	32.86/0.9031	30.44/0.8622	27.18/0.7829
	FFDNet [30]	32.75/0.9029	30.43/0.8641	27.32/0.7906
	DRUNet [51]	32.94/-	30.59/-	27.40/-
	WINNet [38]	32.85/0.9033	30.54/0.8647	27.40/0.7926
	CDNet [36]	32.87/0.9034	30.53/0.8646	27.38/0.7924
	MWDCNN [40]	32.91/0.9037	30.55/0.8651	27.34/0.7914
	MemNet [32]	-	-	27.38/0.7931
	NLRN [52]	33.16/0.9070	30.80/0.8689	27.64/0.7980
	SED-Net	33.06/0.9093	30.72/0.8710	27.55/0.7965
BSD68	BM3D [21]	31.07/0.8717	28.57/0.8013	25.62/0.6864
	WNNM [49]	31.37/0.8766	28.83/0.8087	25.87/0.6982
	TNRD [27]	31.42/0.8769	28.92/0.8093	25.97/0.6994
	NLED [50]	31.43/0.8773	28.93/0.8101	26.02/0.7012
	DnCNN [29]	31.73/0.8907	29.23/0.8278	26.23/0.7189
	FFDNet [30]	31.63/0.8902	29.19/0.8295	26.29/0.7261
	DRUNet [51]	31.79/-	29.31/-	26.36/-
	WINNet [38]	31.70/0.8907	29.27/0.8311	26.36/0.7270
	CDNet [36]	31.74/0.8916	29.28/0.8314	26.36/0.7272
	MWDCNN [40]	31.77/0.8925	29.28/0.8319	26.29/0.7266
	MemNet [32]	-	-	26.35/0.7294
	NLRN [52]	31.88/0.8932	29.41/0.8331	26.47/0.7298
	SED-Net	31.88/0.8995	29.42/0.8403	26.46/0.7348

published methods, which are inspired by or involve wavelet transform, whose comparison can show the benefits of consideration of the whole wavelet analysis model in our method. All comparison results are presented in Table 2.

As the table shows, NLRN and SED-Net win the comparison with the highest PSNR and SSIM. Besides, our SED-Net achieves competitive results compared to NLRN without incorporating nonlocal operations into the network. On BSD68, SED-Net shows the best denoising ability except for only one average PSNR value less than the NLRN's. On Set12, although SED-Net achieves the second-best PSNR, it earns a slightly better SSIM value over NLRN. The result demonstrates SED-Net's superior denoising performance and detail preservation ability. In addition, compared with MWDCNN and WINNet, the proposed method gains obvious improvement, which demonstrates the advantages of the proposed wavelet analysis model-inspired design over just consideration of wavelet transform.

Table 3: Complexity comparison of different Gaussian denoising methods. The number of FLOPs and running time (in seconds) for processing a 256×256 image are reported.

Method	GFLOPs	Time (s)
TNRD [27]	7.99	0.002
DnCNN [29]	36.51	0.013
FFDNet [30]	7.99	0.023
NLRN [52]	1630.30	220.48
MemNet [32]	191.54	0.088
DRUNet [51]	143.48	0.033
WINNet [38]	103.36	0.114
MWDCNN [40]	214.87	0.048
SEDNet	161.26	0.078

Computation time is a critical performance measure for evaluating image denoising methods. In this study, we evaluated the performance of several methods and compared their prediction times, which are presented in Table 3. Among the evaluated methods, SED-Net and NLRN demonstrated the best performance, with SED-Net taking only 0.08 seconds to denoise a 256×256 image, while NLRN required approximately 220 seconds. This is mainly because NLRN incorporates nonlocal operations into neural networks, which increases its runtime consumption significantly. Our proposed method is slightly slower than DnCNN, but its performance is competitive with NLRN, and on average, it increases the peak signal-to-noise ratio (PSNR) by 0.19dB compared to DnCNN.

We also report the giga floating point operations (GFLOPs) for each method in Table 3. For instance, DnCNN with a depth of 17 requires approximately 37 GFLOPs when the input image size is 256×256. In contrast, our proposed method requires approximately 161 GFLOPs, which is roughly 4 times that of DnCNN. Notably, a single nonlocal module in NLRN requires over 100 GFLOPs, which are repeated 15 times during prediction, making NLRN the method with the highest GFLOPs among all compared deep learning methods. Additionally, the trained model of NLRN can only accept patches as input due to the insufficient memory to load the intermediate nonlocal matrix for a large image, resulting in extra patch aggregation. In contrast, DnCNN and our proposed network usually accept the entire image as input.

We also include a *PSNR vs Running Time* plot in Fig. 10, which illustrates the trade-off between the efficiency and effectiveness of the evaluated methods. The

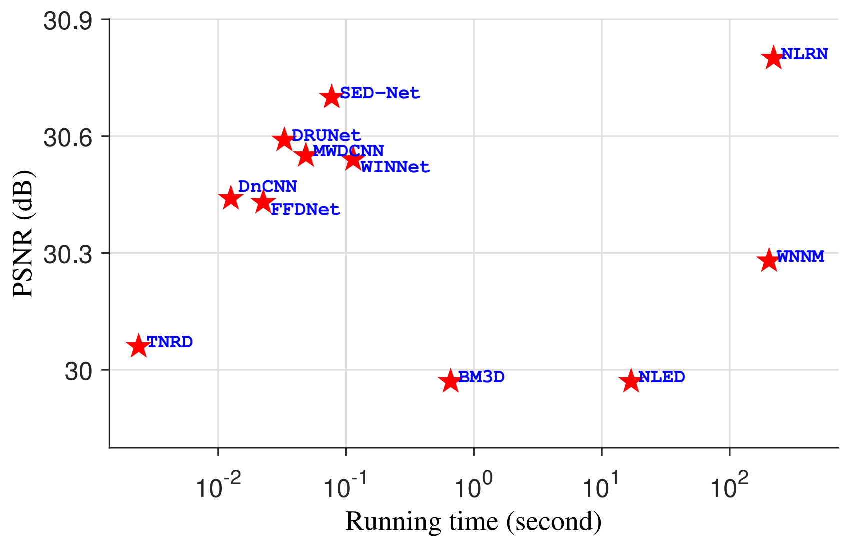
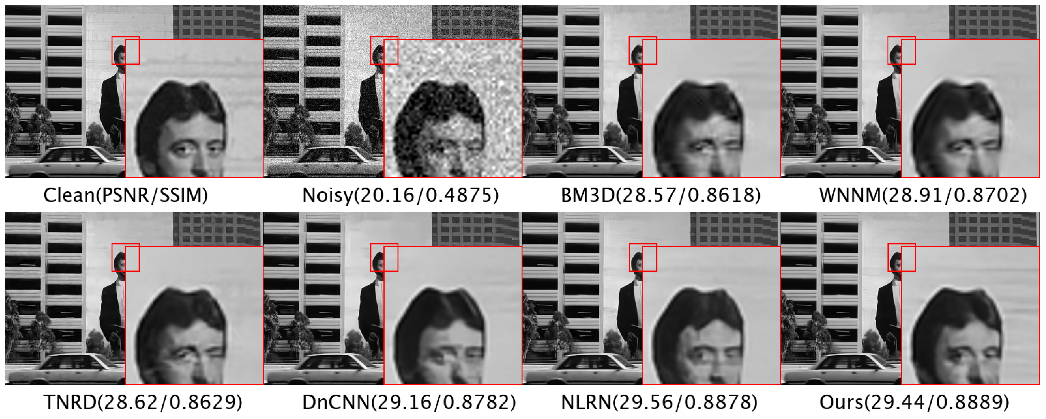


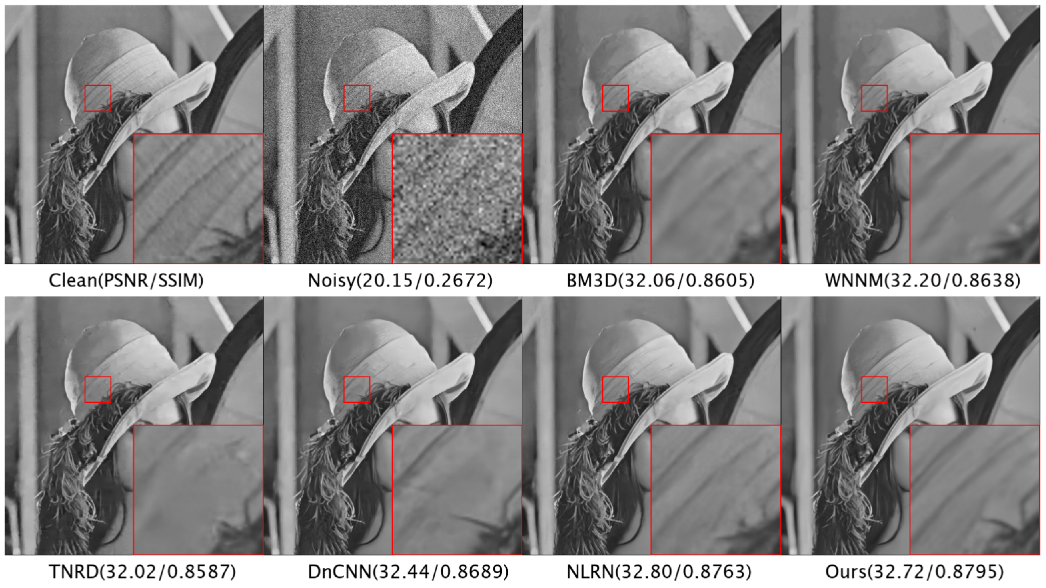
Figure 10: Average PSNR versus computation time on Set12. $\sigma = 25$.

plot shows that SED-Net achieves a good balance between efficiency and effectiveness, providing both state-of-the-art performance and relatively low computational cost.

Putting aside the numbers of PSNR and SSIM, SED-Net also generates better vision results. We show the visual comparison of the SED-Net and other compared methods except for MemNet in Fig. 11. For the huge poster image from BSD68, there are some small edges around the man's head. We zoom in on the area to see the details. It can be seen that only the SED-Net successfully recovers these small edges even though NLRN achieves the best PSNR result. It might be because the compared methods tend to smooth the area around the man's head since there are minor intensity differences between the tiny edges and the background. For methods like BM3D, WNNM, and NLRN that use NSS prior, they may fail to find similar patches correctly due to that the noise corruption is stronger than the edges' sharpness. But our network is capable of enhancing edge preservation via learning directional convolutions. Another image example is the classical Lena image. NLRN and SED-Net recover clear edges of the hat while other methods oversmooth them. Second, it is notable that SED-Net not only suppresses hair's noise but also keeps its right shape compared to other methods, including NLRN.



(a) Huge poster image



(b) Lena image

Figure 11: Visual comparison. The first image is from BSD68 and the second one is Lena image. The noise level is set as 25.

6. Conclusion

This work proposed a CNN-based image denoiser to win the trade-off between computational efficiency and denoising performance. By drawing inspirations from traditional wavelet-based variational models, we developed a model-inspired CNN architecture containing iterative encoding-decoding units, directional convolutions, inception modules, and stage-wise connections. Experimental results showed that our proposed approach achieved high computational efficiency, while still attaining competitive denoising performance compared to state-of-the-art methods. Overall, the proposed method provided a promising solution for efficient image denoising. Our future work will investigate the extension of the proposed directional convolution to a more general patterned convolution so as to design a shallower and faster image restoration CNN with even better performance.

Acknowledgment

Ruotao Xu would like to acknowledge the partial support from National Natural Science Foundation of China under Grant 62106077 and the partial support Natural Science Foundation of Guangdong Province under Grant 2022A1515011087. Yong Xu would like to thank the partial support from National Natural Science Foundation of China under Grant 62072188 and the partial support from State Administration of Foreign Experts Affairs under Grant G2023163015L. Yuhui Quan would like to acknowledge the partial support from National Natural Science Foundation of China under Grant 62372186, the partial support from Natural Science Foundation of Guangdong Province under Grants 2022A1515011755 and 2023A1515012841, and the partial support from Fundamental Research Funds for the Central Universities under Grant x2jsD2230220.

References

[1] Y. Romano, M. Elad, and P. Milanfar. The little engine that could: Regularization by denoising (red). *SIAM J. Imaging Sci.*, 10(4):1804–1844, 2017.

[2] Y. Huang, G. Liao, Z. Zhang, Y. Xiang, J. Li, and A. Nehorai. Sar automatic target recognition using joint low-rank and sparse multiview denoising. *IEEE Geosci. Remote Sens. Lett.*, 15(10):1570–1574, 2018.

[3] Y. Lee, J. Lee, H. Ahn, and M. Jeon. Snider: Single noisy image denoising and rectification for improving license plate recognition. In *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Oct 2019.

[4] J. Yuan. MRI denoising via sparse tensors with reweighted regularization. *Appl. Math. Model.*, 69:552–562, May 2019.

[5] D. Liu, B. Wen, X. Liu, Z. Wang, and T. S. Huang. When image denoising meets high-level vision tasks: a deep learning approach. In *Proc. 27th Int. Jt. Conf. Artif. Intell.ligence*, pages 842–848. AAAI Press, 2018.

[6] S. Lefkimmiatis. Non-local color image denoising with convolutional neural networks. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 3587–3596, 2017.

[7] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu. Residual dense network for image restoration. *arXiv preprint arXiv:1812.10477*, 2018.

[8] D. Yang and J. Sun. BM3D-Net: A convolutional neural network for transform-domain collaborative filtering. *IEEE Signal Process. Lett.*, 25(1):55–59, 2017.

[9] A. Danielyan, V. Katkovnik, and K. Egiazarian. Bm3d frames and variational image deblurring. *IEEE Trans. Image Process.*, 21(4):1715–1728, 2011.

[10] W. Zuo, L. Zhang, C. Song, and D. Zhang. Texture enhanced image denoising via gradient histogram preservation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 1203–1210, 2013.

[11] J.-F. Cai, H. Ji, Z. Shen, and G.-B. Ye. Data-driven tight frame construction and image denoising. *Appl. Comput. Harmon. Anal.*, 37(1):89–105, 2014.

[12] D. Geman and C. Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Trans. Image Process.*, 4(7):932–946, 1995.

[13] S. Mallat. *A wavelet tour of signal processing*. Academic Press, 1999.

[14] A. Beck and M. Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *IEEE Trans. Image Process.*, 18(11):2419–2434, 2009.

[15] L. Tan, W. Liu, and Z. Pan. Color image restoration and inpainting via multi-channel total curvature. *Appl. Math. Model.*, 61:280–299, September 2018.

[16] H. Pan, Y.-W. Wen, and H.-M. Zhu. A regularization parameter selection model for total variation based image noise removal. *Appl. Math. Model.*, 68:353–367, April 2019.

[17] B. Zhang, G. Zhu, Z. Zhu, and S. Kwong. Alternating direction method of multipliers for nonconvex log total variation image restoration. *Appl. Math. Model.*, 114:338–359, February 2023.

[18] R. Yan, L. Shao, and Y. Liu. Nonlocal hierarchical dictionary learning using wavelets for image denoising. *IEEE Trans. Image Process.*, 22(12):4689–4698, 2013.

[19] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Process.*, 15(12):3736–3745, 2006.

[20] C. Bao, H. Ji, Y. Quan, and Z. Shen. Dictionary learning for sparse coding: Algorithms and convergence analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(7):1356–1369, 2015.

[21] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Trans. Image Process.*, 16(8):2080–2095, 2007.

[22] N. Muhammad, N. Bibi, M. A. Shah, S. Zainab, I. Ullah, and Z. Mahmood. An entropy based salient edge enhancement using fusion process. *Appl. Math. Model.*, 93:525–537, May 2021.

[23] M. Zontak, I. Mosseri, and M. Irani. Separating signal from noise using patch recurrence across scales. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 1195–1202, 2013.

[24] Y. Li, W. Dong, G. Shi, and X. Xie. Learning parametric distributions for image super-resolution: Where patch matching meets sparse coding. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 450–458, 2015.

[25] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 479–486. IEEE, 2011.

[26] U. Schmidt and S. Roth. Shrinkage fields for effective image restoration. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 2774–2781, 2014.

[27] Y. Chen and T. Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1256–1272, 2016.

[28] H. C. Burger, C. J. Schuler, and S. Harmeling. Image denoising: Can plain neural networks compete with bm3d? In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 2392–2399. IEEE, 2012.

[29] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Trans. Image Process.*, 26(7):3142–3155, 2017.

[30] K. Zhang, W. Zuo, and L. Zhang. FFDNet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Trans. Image Process.*, 27(9):4608–4622, 2018.

[31] T. Plötz and S. Roth. Neural nearest neighbors networks. In *Adv. Neural Inf. Process. Syst.*, 2018.

[32] Y. Tai, J. Yang, X. Liu, and C. Xu. Memnet: A persistent memory network for image restoration. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 4539–4547, 2017.

[33] S. Yu, B. Park, and J. Jeong. Deep iterative down-up cnn for image denoising. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, pages 0–0, 2019.

[34] B. Park, S. Yu, and J. Jeong. Densely connected hierarchical network for image denoising. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, pages 0–0, 2019.

[35] F. Fang, J. Li, Y. Yuan, T. Zeng, and G. Zhang. Multilevel edge features guided network for image denoising. *IEEE Trans. Neural Netw. Learn. Syst.*, 32(9):3956–3970, 2020.

[36] Y. Quan, Y. Chen, Y. Shao, H. Teng, Y. Xu, and H. Ji. Image denoising using complex-valued deep cnn. *Pattern Recognit.*, 111:107639, 2021.

[37] J. Li, H. Yang, Q. Yi, F. Fang, G. Gao, T. Zeng, and G. Zhang. Multiple degradation and reconstruction network for single image denoising via knowledge distillation. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 558–567, 2022.

[38] J.-J. Huang and P. L. Dragotti. Winnet: Wavelet-inspired invertible network for image denoising. *IEEE Trans. Image Process.*, 31:4377–4392, 2022.

[39] P. Liu, H. Zhang, W. Lian, and W. Zuo. Multi-level wavelet convolutional neural networks. *IEEE Access*, 7:74973–74985, 2019.

[40] C. Tian, M. Zheng, W. Zuo, B. Zhang, Y. Zhang, and D. Zhang. Multi-stage image denoising with the wavelet transform. *Pattern Recognit.*, 134:109050, 2023.

[41] Z. Shen. Wavelet frames and image restorations. In *Proc. Int. Congr. Mathematicians*, pages 2834–2863. World Scientific, 2010.

[42] B. Dong, Z. Shen, et al. Mra-based wavelet frames and applications. *IAS Lecture Notes Series*, 19, 2010.

[43] A. Ron and Z. Shen. Affine systems inl 2 (r d) ii: Dual systems. *J. Fourier Anal. and Applicat.*, 3(5):617–637, 1997.

[44] J.-L. Starck, M. Elad, and D. L. Donoho. Image decomposition via the combination of sparse representations and a variational approach. *IEEE Trans. Image Process.*, 14(10):1570–1582, 2005.

[45] H. Zhang and V. M. Patel. Convolutional sparse coding-based image decomposition. In *Proc. British Machine Vision Conf.*, 2016.

[46] S. Gu, D. Meng, W. Zuo, and L. Zhang. Joint convolutional analysis and synthesis sparse representation for single image layer separation. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 1717–1725. IEEE, 2017.

[47] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. IEEE Int. Conf. Comput. Vis.*, volume 2, pages 416–423. IEEE, 2001.

[48] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[49] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with application to image denoising. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 2862–2869, 2014.

[50] X. Yang, Y. Xu, Y. Quan, and H. Ji. Image Denoising via Sequential Ensemble Learning. *IEEE Trans. Image Process.*, 29:5038–5049, 2020.

[51] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2021.

[52] D. Liu, B. Wen, Y. Fan, C. C. Loy, and T. S. Huang. Non-local recurrent network for image restoration. In *Adv. Neural Inf. Process. Syst.*, pages 1673–1682, 2018.