TSDN: Two-Stage Raw Denoising in the Dark

Wenshu Chen, Yujie Huang, Mingyu Wang, *Member, IEEE*, Xiaolin Wu, *Fellow, IEEE*, and Xiaoyang Zeng, *Member, IEEE*

***Abstract*— Denoising is one of the most significant procedures in the image processing pipeline. Nowadays, deep-learning-based algorithms have achieved superior denoising quality than traditional algorithms. However, the noise becomes severe in the dark environment, where even the SOTA algorithms fail to achieve satisfactory performance. Besides, the high computational complexity of deep-learning-based denoising algorithms makes them hardware unfriendly and difficult to process high-resolution images in real-time. To address these issues, a novel low-light RAW denoising algorithm Two-Stage-Denoising (TSDN), is proposed in this paper. In TSDN, denoising consists of two procedures: noise removal and image restoration. Firstly, in the noise-removal stage, most noise is removed from the image, and an intermediate image that is easier for the network to recover the clean image is obtained. Then, in the restoration stage, the clean image is restored from the intermediate image. The TSDN is designed to be light-weight for real-time and hardware friendly. However, the tiny network will be insufficient for satisfactory performance if directly trained from scratch. Therefore, we present an Expand-Shrink-Learning (ESL) method to train the TSDN. In the ESL method, firstly, the tiny network is expanded to a larger one with similar architecture but more channels and layers, which enhances the learning ability of the network because of more parameters. Secondly, the larger network is shrunk and restored to the original small network in fine-grained learning procedures, including Channel-Shrink-Learning (CSL) and Layer-Shrink-Learning (LSL). Experimental results demonstrate that the proposed TSDN achieves better performance (PSNR and SSIM) than other SOTA algorithms in the dark environment. Besides, the model size of TSDN is one-eighth of that of the U-Net for denoising (a classical denoising network).**

***Index Terms*— Image processing, denoising, light-weight network.**

Manuscript received 26 October 2022; revised 8 May 2023; accepted 19 June 2023. Date of publication 28 June 2023; date of current version 10 July 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62074041, in part by the State Key Laboratory of ASIC and System under Grant 2021KF009, in part by the Zhuhai Fudan Innovation Institute, and in part by the Key Research and Development Program of Shandong Province under Grant 2022CXGC010504. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Diego Valsesia. *(Wenshu Chen and Yujie Huang contributed equally to this work.) (Corresponding authors: Mingyu Wang; Yujie Huang.)*

Wenshu Chen and Yujie Huang are with the State Key Laboratory of ASIC and System, College of Microelectronics, Fudan University, Shanghai 200000, China, and also with Shanghai ExploreX Technology Company Ltd., Shanghai 200120, China (e-mail: chenws21@m.fudan.edu.cn; huangyj19@fudan.edu.cn).

Mingyu Wang and Xiaoyang Zeng are with the State Key Laboratory of ASIC and System, College of Microelectronics, Fudan University, Shanghai 200000, China (e-mail: mywang@fudan.edu.cn; xyzeng@fudan.edu.cn).

Xiaolin Wu is with the Department of Electrical and Computer Engineering, McMaster University, Hamilton, ON L8G 4K1, Canada (e-mail: xwu@ece.mcmaster.ca).

Digital Object Identifier 10.1109/TIP.2023.3289049

I. Introduction

DENOISING aims to suppress noise in the image and is dominant in image processing. It is critical not only for aesthetic purposes but also for other downstream tasks, such as detection [3]. Given a noise-free image I and the noise corrupting n, the observed noisy image $\hat{I}$ can be expressed as:

$$\hat{I} = I + n \tag{1}$$

Denoising is responsible for restoring I from $\hat{I}$, including the research areas of image noise modeling and reduction [2], [4], [5], [6].

In good light conditions, the real-world noise in the image can be modeled as Poissonian-Gaussian noise [7]. In these conditions, the traditional denoising algorithms: Non-Local Means (NLM) [5] and BM3D [6], can achieve satisfactory quality. However, when the light is insufficient in the environment, where the signal-to-noise ratio drops sharply and noise floods the image, the traditional denoising algorithms can no longer guarantee the quality. This situation is worse for smartphone cameras due to their high resolution with a small photosensitive area, making sufficient exposure more difficult.

Thanks to the development of deep learning, the quality of denoising ushers in an explosive increase [8], [9], [10], [11], [12], [13]. These algorithms usually consist of deep neural networks and learn the statistical regularities from the datasets, which contain noisy images and their clean counterparts. Existing deep-learning-based (DL-based) algorithms mainly focus on RGB domain denoising and employ the RGB images of the most popular datasets, SIDD [14] and DnD [15], as training and benchmark data. And it is known that the RGB images of the camera are obtained by two hardware imaging procedures: firstly, the image sensor converts the photon signal electrical signal and obtains RAW image; secondly, the image signal processor (ISP) processes the RAW data with several procedures (such as denoising [5] and demosaicing [16]) and obtains the finally visual-friendly RGB images.

However, the ISP pipeline will introduce additional signal-dependent and spatial-dependent noise, which makes the noise distribution more complex, and its effect becomes more severe in the dark environment because the original noise distribution is extremely sophisticated and the standard deviation of the noise is large. Therefore, it is wiser to denoise in the RAW domain. However, in a dark environment, the SOTA DL-based algorithms cannot restore the image well.

The most typical low-light RAW denoising dataset is SID [1], which contains noisy images and their "clean" counterparts. Note that in this paper, all visual images are obtained

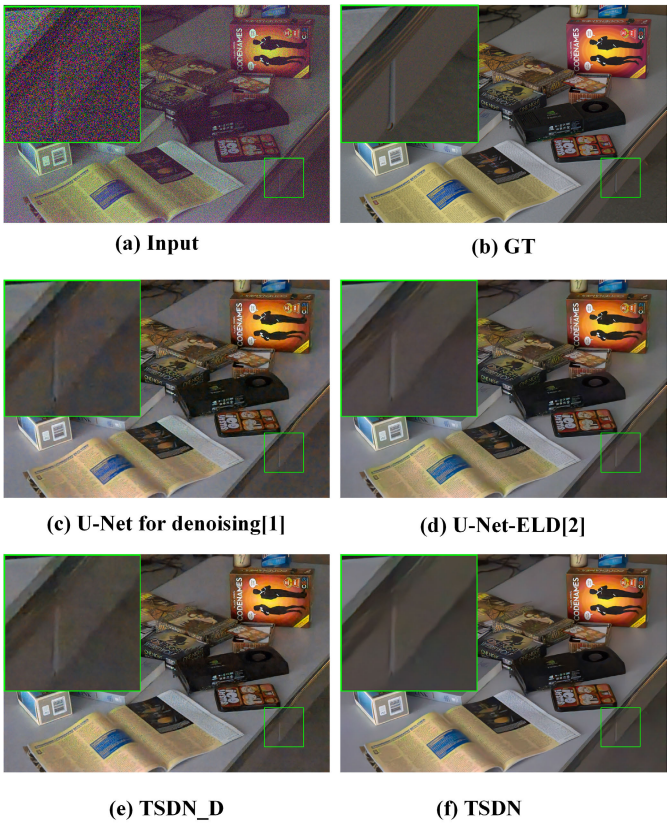


Fig. 1. The noisy image in the dark (a) and the ground truth (b). The result of classical denoising network U-Net [1] is in (c) and its optimized result [2] is in (d), which addresses the low light denoising by dataset and noise parameter calibration. Denoising in the dark is modeled as noise removal and image restoration in the proposed TSDN. The first stage of TSDN (TSDN_D) is responsible for noise removal and the output image is in (e), in which most of the noise is removed. And the final processed image of TSDN is shown in (f), in which the image is restored and enhanced by the restoration stage.

by interpolation of RAW images. And the quantitative results (e.g., SSIM and PSNR) are obtained in the RAW domain.

U-Net [17] is a classical and widely-used deep neural network first proposed for image segmentation [17]. In recent years, its architecture has been found to be effective for denoising [1]. Based on the architecture of U-Net for image segmentation [17], many denoising algorithms [1], [2], [13], [18], [19] are proposed and have achieved impressive quality on real noisy images after being trained on the well-synthetic or paired real datasets [1], [2]. In the rest of the paper, "U-Net" refers to the U-Net for denoising in [1]. When the environment is dark, U-Net [1] fails to achieve satisfactory results. In U-Net [1], cross-layer connections are utilized to recover the image of full resolution. This mechanism is effective when the noise level of the input image is low. However, when the noise is severe, the cross-layer feature maps that contain much noise will decrease the performance. As shown in Figure 1 (c), there are 'scars' left in the noise-intensive dark areas. Wei et al. [2], [18] address the issue of denoising in the dark from the perspective of the dataset. They proposed a low-light noise model and used this model to generate a dataset (ELD dataset). Networks pre-trained on this dataset can better complete the task of low light denoising. As shown in Figure 1 (d), the denoising performance is improved. However, their training dataset is not publicly available due to commercial reasons.

Besides, the enormous computational complexity and model size of the deep neural networks [8], [9], [10], [11], [12], [13] make them hardware unfriendly and hard to process high-resolution images in real-time. Although The existing pruning [20], [21], [22], [23] and quantization [24], [25], [26], [27] technologies can make them hardware friendly to a certain extent, their original enormous computational complexity and model size limit their potential.

To address the above problems, we propose a novel algorithm for RAW denoising in the dark environment named TSDN, where denoising consists of two procedures: noise removal and image restoration.

Different from [2], we address the issue of denoising in the dark from the perspective of network architecture, which is complementary to the dataset-based solutions. Moreover, the network is carefully designed to solve the problem of large model size. In extremely low-light conditions, severe noise will fill the image (Figure 1(a)). It is challenging for a network to recover the clean image from the noisy image due to poor signal-to-noise ratio (SNR). As shown in Figure 1(c), after denoising of a single-stage network (U-Net for denoising [1]), the remained noise patches blur the image.

Poor SNR brings a large gap between noisy images and clean images, which is the main challenge for denoising in the dark. From this insight, it is wiser to provide an intermediate image for a denoising network to recover the clean image better. Thus we model denoising in the dark as two stages: noise removal and image restoration. Firstly, in the noise removal stage, most noise is removed from the image and an intermediate result is obtained (Figure 1(e)). Then, the image is restored in the restoration stage and a clean image is achieved (Figure 1(f)).

The TSDN is designed to be tiny for real-time and hardware friendly. However, because of the small number of trainable parameters, the learning capacity of small networks is limited, which will lead to unsatisfactory results if trained from scratch. Therefore, we present an Expand-Shrink-Learning (ESL) method to train the TSDN.

The core idea of ESL is to improve the performance of a small network by leveraging the powerful learning capability of a large network. In the Expand-Shrink-Learning method, the first phase is "expand" where the tiny network is expanded to a larger one with similar architecture but more channels and layers, which endow the network with stronger learning capacity because of more trainable parameters. And the second phase is "shrink" where the larger network is shrunk and restored to the original small network with fine granularity. After ESL, the performance of TSDN outperforms U-Net with a large margin with the model size one-eighth of that of the U-Net for denoising [1].

The main contributions of our work are:

1) A novel denoising algorithm Two-Stage-Denoising (TSDN) is proposed. In the TSDN, denoising is modeled as two procedures: noise removal and image restoration.
2) We present an Expand-Shrink-Learning (ESL) method that can improve the performance of tiny networks.

3) The experimental results demonstrate that the TSDN achieves better performance than existing SOTA algorithms in the dark environment with smaller model size.

The rest of the paper is organized as follows. The related work is introduced in Section II, while the proposed TSDN and ESL are described in detail in Section III. Then, the experimental results are discussed in Section IV. Finally, the work is concluded in Section V.

II. Related Work

A. Traditional Denoising Algorithms

Traditional methods are proposed based on prior assumptions about the noise distribution (e.g., Gaussian distribution). A simple traditional denoising method is the sliding mean filter [28], which replaces each pixel with the mean of its neighboring pixels. To avoid the influence of irrelevant neighbors, Tomasi and Manduchi [29] proposed a bilateral filter that averages neighbors according to color and spatial similarities. Non-Local-Means (NLM) [5] is the most classical traditional denoising algorithm that replaces a pixel with a weighted average of all pixels in the image. Based on the idea of NLM and using block-matching and 3D filtering, a more complicated denoising algorithm BM3D [6] is proposed and achieves the SOTA results among the traditional algorithms.

B. Dl-Based Denoising Algorithms

Although traditional algorithms [5], [6] can perform well with Gaussian noise, they fail to achieve satisfactory results in real-world denoising. In recent years, DL-based denoising algorithms have become a research hotspot and many remarkable results in real-world denoising have been reported [1], [2], [13], [18], [19]. The DL-based algorithms can be generally classified into RAW-based approaches and RGB-based approaches. RGB-based denoising approaches are the most studied. Whether RGB denoising or RAW denoising, the core idea is to model the noise characteristics and then remove the noise. Therefore, by modifying the network parameters and changing the training set, the RGB denoising networks [13], [19], [30] can also be migrated to RAW denoising. Guo et al. [19] propose a convolutional blind denoising network (CBDNet), which has a noise estimation subnetwork and is trained with real-world noisy-clean image pairs. Cheng et al. [13] propose NBNet, which utilizes subspace projection and self-attention mechanism and achieves SOTA performance on SIDD dataset [14]. Zha et al. [31], [32], [33], [34], [35], [36] propose to use low-rank-based methods for image restoration.

RAW-based algorithms [1], [2], [18], [37], [38], [39] have attracted more attention recently. Eli et al. [37] propose DeepISP to simulate real ISP, which contains joint denoising and demosaicing. Chen et al. [1] propose the SID dataset and use U-Net [17] for low-light RAW denoising. Lu and Jung [38] propose a low-light imaging framework that performs joint illumination adjustment, color enhancement, and denoising. Wei et al. [2], [18] propose a noise formation model and a noise parameter calibration method for extreme low-light denoising, which addresses the issue of denoising from the perspective of dataset and achieves SOTA denoising performance in the dark environment.

DL-based algorithms are data-driven. Therefore, high quality datasets are crucial for DL-based denoising algorithms. Some researches have resorted to collecting paired real data (clean and noisy images) for evaluation and training [1], [14], [37], [40], [41]. SIDD [14] and DnD [15] are the most popular RGB denoising training and benchmark datasets. And the most popular low-light RAW denoising dataset is SID [1]. However, collecting the real noisy images and their clean counterparts is tremendously labor-intensive and expensive. Hence, other researchers try to synthesize the noisy data [2], [4], [7], [42], [43]. They add artificial noise to the clean images to obtain the noisy images, and then the dataset contains clean images and their noisy counterparts. The most commonly used ones are the homoscedastic or heteroscedastic Gaussian noise models [7]. To make the synthesized noise closer to the real-world noise, some works [2], [2], [18] try to model the noise during the real imaging process, which improves the realism of synthetic training data and enhances the performance of the deep neural networks on real noisy images.

C. Light-Weight Methods

The typical approach to obtain a light-weight network is model compression [44], such as knowledge distillation [45], [46], [47], [48]. The core idea of knowledge distillation is transferring the knowledge learned by a network (teacher network) to another network (student network) [45]. The training process of the student network requires the involvement of the teacher network and a knowledge distillation loss function is necessary [46]. Different from knowledge distillation, in the proposed ESL method, the current network utilizes the expanded network's parameters for initialization and is trained on the dataset alone without the teacher network and knowledge distillation loss. Another related work is [49], where the authors proposed to expand each linear layer of a compact network into a succession of multiple linear layers without any non-linearity in between. They only expand linear layers to multiple linear layers since consecutive linear layers are equivalent to single one. However, the non-linear learning ability is crucial for CNN and their method can not improve the non-linearity of the networks. Different from [49], in ESL, the expanded layers can be either linear or non-linear, which can significantly improve the learning ability of the network.

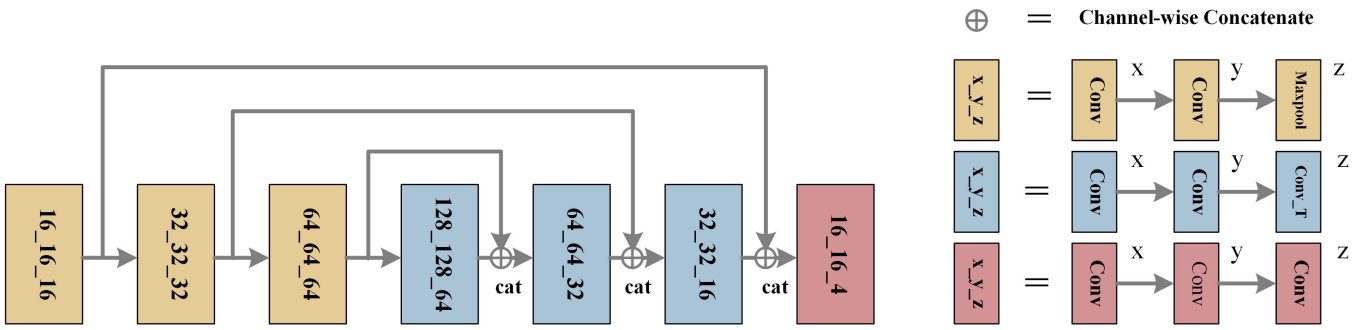
III. Method

In this section, Firstly, the noise model in the dark environment is analyzed. And then the two-stage denoising network (TSDN) is presented and the Expand-Shrink-Learning (ESL) method is described in detail.

A. Analysis

The noise in the imaging process consists of signal-dependent noise and signal-independent noise. In good light conditions, the signal is strong, so the other signal-independent noise that is non-Gaussian distribution (such as read-out

(a) The two-stage denoising procedure



(b) The architecture of TSDN_D and TSDN_R

Fig. 2. The architecture of the proposed two-stage denoising algorithm. In the first stage, the input noisy image is processed by TSDN_D, which is responsible for denoising. We can see that the image becomes blurred and there still remains some noise. And in the second stage, the output image of the first stage is processed by TSDN_R, which is responsible for the restoration. After two-stage processing, the image can be recovered from the noisy image. The architecture of TSDN_D and TSDN_R is shown in (b), each contains three forms of Convolution Blocks (ConvBlocks): (1)two convolution (Conv) layers with one max pooling layer (Maxpool); (2)two convolution layers with one transpose convolution layer (Conv_T);(3) three convolution layers. They are illustrated in the bottom right corner and the number after each layer (**x, y, and z**) is the **output channel number**. Note that all the kernel size of the convolution layers is 3×3 and the stride is 1. The kernel size of the pooling layer is 2×2 and the stride is 2. The kernel size of the transpose convolution layer is 2×2 and the stride is 2. Besides, the number x_y_z in each block means the output channel number of each layer.

noise [18] of the circuit) can be ignored. In this situation, the Poissonian-Gaussian distribution model [7] can provide a good approximation of the real noise. When the environment is dark and the signal is weak, the Poissonian-Gaussian noise model [7] can be calibrated as:

$$z(x) = y(x) + \eta_p(y(x)) + \eta_g(x) + \xi(x) \tag{2}$$

where $x \in X$ is the pixel position domain, $y : x \in X \rightarrow y \in R$ is the original signal (unknown), $\eta_p(y(x))$ is a signal-dependent noise component, $\eta_g(x)$ is a signal-independent noise component that obeys Gaussian distribution, and $\xi(x)$ is signal-independent noise component that is non-Gaussian distribution.

In the dark environment, the noise becomes so severe that the image content is almost covered by noise. That is, the signal-independent noise $\eta_p(y(x))$ described in the above equation is equal to or even larger than the signal $y(x)$. In this condition, it is very challenging for a network to directly recover the real image from the noisy image because of the large gap between them.

In deep learning (DL), "depth" brings a large receptive field, which facilitates the network to extract global features and obtain high-level semantic information. And to reduce computational complexity and eliminate redundant information, downsampling is often used in DL networks. However, downsampling introduces a loss of resolution (pixel) information, which is detrimental to image processing, such as image denoising. Therefore, U-Net [1] uses cross-layer connections, which can pass the feature maps containing resolution and pixel information from the shallow layers of the encoder to the decoder layers to help reconstruct the image. However, as described in Equation 2, the signal-independent noise will have a serious impact on the signal when the environment is dark. In this situation, the cross-layer feature maps that contain pixel information will also contain severe noise, which is detrimental to pixel reconstruction (As shown in Figure 1 (c), there are noise blocks in the image).

In order to avoid a large input-output gap that makes the cross-layer connection affect the reconstruction of the image, we introduce an intermediate result to overload this difference. For example, it is very difficult to change from 0 to 100 because of the large gap, but in two steps, first 0 to 50, and then from 50 to 100 will be easier. Therefore, we propose the two-stage denoising (TSDN) network, which provides a step (intermediate result) between the ground truth and the noisy image. The intermediate image reduces the adverse effects of the large gap between the input and output of the network. From another perspective, the first stage of TSDN is responsible for removing most of the noise, which improves the quality of the second stage TSDN cross-layer feature maps and contributes to image restoration in the second stage. The experimental results show that the two-stage

strategy is effective and achieves better results in low-light conditions.

B. Two Stage Denoising Network

The idea of stage-by-stage processing has been described in Section III-A. To balance the computational complexity and performance of the network, we do experiments on the stage number and find that two-stage is sufficient on the existing extremely low light denoising task–SID dataset [1] ((TSDN4 vs. ThSND in Table II)). The proposed two-stage denoising procedure and the architecture of the network are shown in Figure 2. The two stages, TSDN_D and TSDN_R have the same compact cross-layer architecture. Each subnetwork contains three forms of Convolution Blocks (ConvBlocks): (1) two convolution layers with one max pooling layer; (2) two convolution layers with one transpose convolution layer;(3) three convolution layers. The kernel size of all convolution layers is 3×3 and the stride is 1. Each convolution layer (except for the last layer) is followed by the Leaky-ReLu activation function.

The first stage (TSDN_D) is responsible for noise removal and the second stage (TSDN_R) is responsible for image restoration. In the first stage, most of the noise is removed from the image. After processing of the first stage, the intermediate distribution of the image is obtained. The function of the first stage can be formulated as:

$$I_{mid} = DN(I_{noisy}; \theta) \tag{3}$$

where I_{noisy} is the input image, I_{mid} is the first stage denoised image, and $DN(;\theta)$ denotes the first stage network with parameter θ. Assume an ideal denoising algorithm that can remove all the noise while keep the signal, this can be expressed as:

$$y'(x) = DN_{ideal}(z(x)) = y(x), \forall z(x) \in R \tag{4}$$

where $DN_{ideal}(\cdot)$ is the function of the ideal denoiser, $z(x)$ is the observed noisy image, $y(x)$ is the "clean" (unknown) image, and $y'(x)$ is the denoised image. The true "clean" image is unknown in reality, so the ground truth images of the dataset are regarded as "clean" labels to optimize the network. Thus the loss function (pixel reconstruction loss) for the first stage is:

$$L_{dn} = ||D(I_{noisy}; \theta) - I_{gt}||_2 \tag{5}$$

where I_{gt} is the ground truth image of the dataset.

As for the second stage, it is responsible for image restoration, that is, mapping the intermediate image (distribution) to clean image (distribution). This stage can be formulated as:

$$I_{dn} = R(I_{mid}; \omega) \tag{6}$$

where I_{dn} is the denoised image, I_{mid} is the first stage output image, and $R(;\omega)$ denotes the second stage network with parameter ω. The loss function (restoration loss) for the second stage is:

$$L_{rs} = ||R(I_{mid}; \omega) - I_{gt}||_2 \tag{7}$$

The first stage and the second stage are trained together for collaborative optimization, thus the total loss is:

$$L_{total} = \alpha \cdot L_{dn} + \beta \cdot L_{rs} \tag{8}$$

where α and β are hyper-parameters to balance the two loss. In this paper, both the values of α and β are set to 1.

C. The Expand-Shrink-Learning (ESL) Method

To realize fast image processing, we design a tiny denoising network TSDN, which is described in section III-B and the model size is only 3.8 MB. The model size of TSDN is only $\frac{1}{8}$ of that of the popular network U-Net for denoising [1]. This brings a great challenge for training the network, and the performance will not be satisfactory if it is trained directly on the dataset from scratch (see the experimental results section). Besides, in recent years, DL-based neural networks have shown a tendency to increase in size. Large-scale parameters bring powerful learning capability to DNNs. And the "large" is reflected in the number of channels and convolution layers. More channels mean the network can extract more features and more layers mean that the network can extract deep features. From this insight, we propose the Expand-Shrink-Learning method to train the small network and help it improve performance. There are two main procedures in ESL.

As shown in Figure 3, First, the convolution layers and channels of the original network (Figure 3 (a)) are expanded to get a large network (Figure 3 (b)). Then the network is trained on the dataset and the step is called Expand-Learning (EL). The expansion rule is that increase channel numbers and layers while keeping the original feature architecture. For example, the feature architecture of TSDN is encoder-decoder based with cross-layer connection and feature map concatenating. Thus, firstly, the channels are doubled, which helps the network extract more features. And then, the middle layers with more channels are added and thus one more cross-layer connection is introduced. The reason for adding middle layers is that TSDN_D and TSDN_R are encoder-decoder based architecture, so the middle layers are the deepest layers in the network, and adding deep layers helps the network extract more deep features.

Second, the large network is shrunk to the original network by reducing the channels (Channel-Shrink-Learning, CSL) and layers (Layer-Shrink-Learning, LSL), which are fine-grained procedures. Note that every time the network is modified (shrunk), it will be retrained on the dataset. As shown in Figure 3, in the first step of the CSL, the channel numbers of the middle layers of (b) are halved, and the current network (Figure 3 (c)) is trained on the dataset using parameters of the large network (which has been trained in the last step, Figure 3 (b)) for initialization. And then channels of the layers nearing the middle layers are halved to maintain the proportion between channels of different layers, which is illustrated in Figure 3 (c) and (d). Repeat this until all channels (except for input and output channels of the network) are halved, which is illustrated in Figure 3 (d) and (e). After that, as shown in Figure 3 (e) and (f), cut back the middle layers to retrieve the original network (LSL). In the steps of LSL, trained

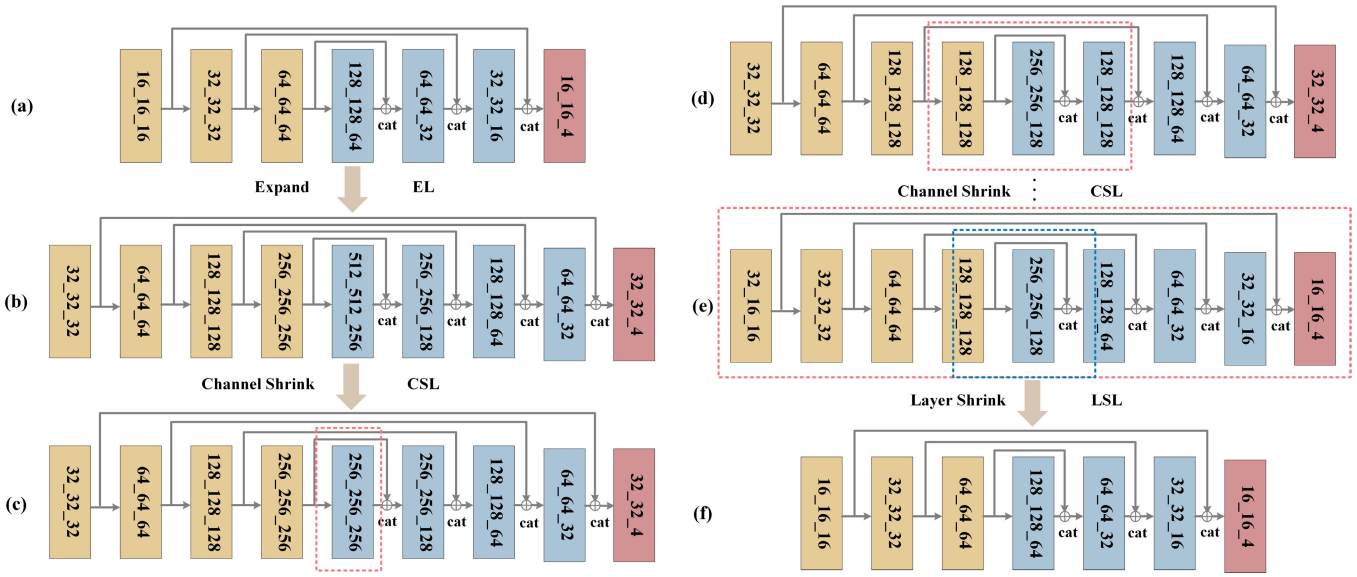


Fig. 3. The illustration of the proposed ESL method. The network (a) is first expanded into a large network (b) by increasing the channels and layers and then is trained on the dataset. Then, the network is shrunk to the original architecture through the shrink-learning process, in which the number of channels is halved (shrink channels, CSL) step by step until all the channels are halved (except for the input and output channels), and then the middle layers of the network are removed (shrink layers, LSL). The blocks in the red dashed box refer that their channels are halved. And the blocks in the blue dashed box refer that the layers are removed (the last 2 layers of the first ConvBlock and the first layer of the third ConvBlock are removed and the remained parts of the 2 ConvBlocks are fused to one ConvBlock). Note that the networks above refer to both TSDN_D and TSDN_R, and are retrained with **INIT1** and **INIT2** once modified in ESL.

parameters of the large network (which has been trained in the last step) are also employed for initializing the parameters of the small network.

For the steps of reducing the channels (CSL), each current network is initialized with the trained parameters of the last step network. The initialization (**INIT1**) can be expressed as:

$$\begin{aligned} &\Theta_{cur}(l, x^l, y^l, z) = \Theta_{last}(l, x^l, y^l, z), \\ &l \in [0, L-1], x^l \in [0, N_o^l - 1], \\ &y^l \in [0, N_i^l - 1], z \in [0, 8] \end{aligned} \tag{9}$$

where Θ_{cur} denotes the parameters of the current network to be trained; Θ_{last} denotes the last step network that has been trained in the last step; l denotes the layer index and L is the total layer number of the networks; x^l denotes the output channel index of the $l-th$ layer and N_o^l is the output channel number of layer l in the current network; y^l denotes the input channel index of the $l-th$ layer and N_i^l is the input channel number of layer l in the current network; z denotes the index of one 3×3 convolution kernel. In each step, 2 or 4 layers are performed with channel halving in the channel shrink procedure (CSL), which makes the small network more similar to the large network and better to inherit the knowledge from the large network through **INIT1**.

For the steps of reducing the layers (LSL), the layers remained in the current network use the same parameters of the last network for initialization. Suppose that the layers l_i to l_j are removed in the last network to get the current network, this initialization (**INIT2**) can be expressed as:

$$\begin{aligned} &\Theta_{cur}(l, x^l, y^l, z) = \Theta_{last}(l, x^l, y^l, z), \\ &\qquad l \in [0, l_i) \cup (l_j, L-1], x^l \in [0, N_o^l - 1], \\ &\qquad y^l \in [0, N_i^l - 1], z \in [0, 8] \end{aligned} \tag{10}$$

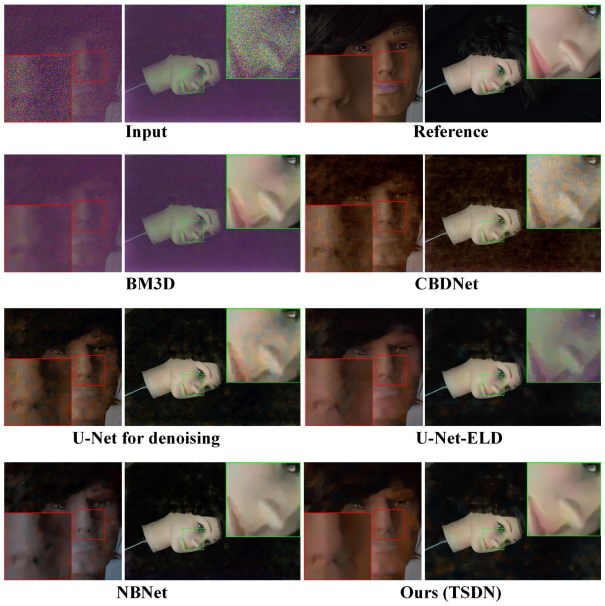


Fig. 4. The visual quality comparison of our algorithm and other SOTA algorithms: BM3D [6], U-Net [1], CBDNet [19], U-Net-Eld [2], and NBNet [13] (recommend to zoom in).

where Θ_{cur} and Θ_{last} denote the parameters of the current network and last network; N_o^l and N_i^l denote the output channel number of layer l in the current (last) network. And to minimize the structural gap between the large network and the small network, two layers are removed at every step.

IV. Experimental Results

A. Implementation Details

We use the widely used SID Sony dataset [1] to train (training set) and evaluate (test set) all the networks

TABLE I
QUANTITATIVE RESULTS ON SONY SID DATASET [1]

Models	×100		×250		×300		Model Size (MB)	GFLOPs	time (ms)
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM			
BM3D [6]	32.92	0.758	29.56	0.686	28.88	0.674	#	#	#
U-Net for denoising [1]	38.56	0.906	36.40	0.870	35.56	0.854	29.6	96.77	10.96
CBDNet [19]	38.12	0.893	35.70	0.842	34.84	0.818	16.7	297.63	23.79
U-Net-Eld [2]	38.67	0.914	37.35	0.891	36.63	0.881	29.6	96.77	10.96
NBNet [13]	39.02	0.912	36.58	0.879	35.54	0.866	50.8	216.44	#
DeamNet [30]	39.09	0.916	37.14	0.890	36.45	0.880	8.6	429.95	129.84
Ours	38.75	0.914	37.37	0.892	36.69	0.883	3.8	37.31	8.33

The other SOTA models are trained and evaluated using the same settings described in section IV-A. Note that $\times n$ in the first line denotes the exposure time ratio of the ground truth and noise image. And the larger number means that the environment is darker and the noise is more severe. GFLOPs and inference time are computed on a 24 GB NVIDIA TITAN RTX GPU when the size of RAW image is 1024×1024. The best results in the table are in the color red and the second-best are in the color blue (other tables are the same with this).

(including the traditional and deep learning ones) in this section. SID Sony dataset [1] contains noisy RAW images and their counterparts, and the noisy RAW images are captured in the dark environment, which brings severe noise. The evaluation method is the same as [18], in which 15 indoor scenes (the noise is extremely severe) are selected from SID dataset for evaluation. The images in the SID dataset are high-resolution Raw Bayer images (4256×2848), so the raw Bayer images are packed into four channels (R-G-B-G) and non-overlapped 512×512 regions are cropped during training. The cropped images are augmented by random flipping or rotation. The networks are trained with 300 epochs and batch size 1. During the training, L_1 loss and Adam [50] optimizer are applied. Initially, the learning rate is 10^{-4}, and then it is halved at epoch 150. Finally, after 250 epochs, it is reduced to 10^{-5}.

B. Quantitative and Qualitative Comparison

The quantitative results (PSNR and SSIM) of our TSDN and other SOTA models are shown in Table I. When the noise level is low (×100), TSDN achieves better results than BM3D [6], U-Net [1], U-Net-Eld [2], and CBDNet [19], and little inferior to NBNet [13] and DeamNet [30], which are designed for denoising in normal lighting conditions. And it can be concluded from Table I that when the noise is more severe (×250 and ×300, extremely low-light conditions), the proposed TSDN achieves the best results. Besides, the model size and computational complexity of TSDN are much smaller than the other SOTA models, which means it is more efficient. Note that when the image size is 1024×1024, the GPU has not enough memory for NBNet [13]. Besides, since the computation of GPU is parallel, the inference time is not necessarily proportional to computational complexity. Combining the above results, the proposed TSDN achieves SOTA performance in low-light conditions with smaller network complexity.

The visual comparison is shown in Figure 4, and it can be found that the proposed TSDN achieves more satisfactory performance than the superior traditional algorithm BM3D [6] and other SOTA DL-based algorithms: U-Net [1],U-Net-Eld [2], BCDNet [19], and NBNet [13]. After processing by BM3D [6], U-Net in [1], CBDNet [19], there is still obvious residual noise in the resultant images. And the image quality of NBNet [13] is not satisfactory enough because the recovered edges and colors of the image are a bit distorted. Besides, as shown in Figure 4, there is some residual noise in the resultant images of U-Net-ELD [2], while TSDN obtains images with clear details and accurate colors, which means the two-stage denoising strategy is effective.

C. Ablation Study

In this section, The effectiveness of the proposed two-stage denoising algorithm (TSDN), the Expand-Shrink-Learning (ESL) method and the analysis of mean and standard deviation of noise are demonstrated by experiments.

1) Two-Stage: To demonstrate the effectiveness of the two-stage strategy (noise removal for the first stage and image restoration for the second stage), the variable conditions are only related to the two-stage denoising architecture. Thus the expanded network (the second network in Figure 3) is employed and trained for evaluation without the Shrink-Learning (CSL and LSL) procedures. As shown in Table II, it can be concluded that the proposed two-stage denoising algorithm is effective. Firstly, TSDN1, TSDN2, and TSDN4 demonstrate that the two-stage architecture is helpful for denoising. Then, ThSDN vs. TSDN4 illustrates that two-stage is sufficient enough for the task of extremely low-light denoising (SID dataset [1]). And two-stage saves computational complexity and model size compared to more stages. In addition, to directly utilize the existing extremely low light denoising dataset in ThSDN, noisy-clean image pairs in the dataset are used to construct the intermediate loss function, which may not be quite suitable and therefore leads to slight performance loss compared to TSDN. More suitable constraints of intermediate results need to be investigated. Besides, TSDN1 vs. TSDN3 show that if the second stage of TSDN is also responsible for denoising, it will decrease the performance. Finally, TSDN3 and TSDN4 prove that the two stages have different functions, which supports the proposed idea that denoising in the dark can be modeled as noise removal and image restoration.

2) Initialization: In every step of the ESL learning procedure, the current network uses the parameters of the last trained network for initialization. The initialization strategy is provided in Section III-C. The shrinkage process is fine-grained and several network layers are modified while most layers are retained in every step. The unchanged layers' initial parameters are the same as the last network. And for the layers that the channels are reduced, they use the first No and Ni channels' parameters of the last network for initialization. To exploit the influence of the initialization strategies on the modified layers,

TABLE II

ABLATION STUDY OF THE PROPOSED TWO-STAGE DENOISING (NOISE REMOVAL AND IMAGE RESTORATION) ALGORITHM

Models	×100		×250		×300	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
TSDN1	38.56	0.906	36.40	0.870	35.56	0.854
TSDN2	38.79	0.912	37.07	0.887	36.35	0.877
TSDN3	38.29	0.901	36.61	0.877	35.88	0.865
TSDN4	38.99	0.914	37.37	0.892	36.69	0.883
ThSDN	38.91	0.913	37.29	0.887	36.56	0.876

In this table, to demonstrate the effectiveness of the two-stage architecture (noise removal for the first stage and image restoration for the second stage), the variable conditions are only related to two-stage architecture. Thus the expanded network (the second network in Figure 3) is employed for evaluation without the Shrink-Learning (CSL and LSL) procedures. **TSDN1**: single expanded network. **TSDN2**: two-stage architecture, but take the two networks as a whole and are trained only with one loss (pixel reconstruction loss, in equation 5). **TSDN3**: two independently trained networks are cascaded directly. **TSDN4**: two-stage architecture, trained with pixel reconstruction loss and restoration loss (in equation 8). **ThSDN**: three-stage denoising network.

TABLE III

ABLATION STUDY ON DIFFERENT INITIALIZATION STRATEGIES

Models	×100		×250		×300	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
First	38.69	0.914	37.53	0.894	36.78	0.885
Last	38.84	0.914	37.43	0.892	36.72	0.884
Combined	38.57	0.912	37.46	0.891	36.82	0.884

(1) **First**: network k+1 uses the first No and Ni channels' parameters of network k for initialization; (2) **Last**: network k+1 uses the last No and Ni channels' parameters of network k for initialization; (3) **Combined**: network k+1 uses the combination (average) of all channels' parameters of network k for initialization.

TABLE IV

ABLATION STUDY OF THE PROPOSED ESL METHOD

Models	×100		×250		×300		
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	Model Size (MB)
TSDN_CSL1	38.56	0.910	37.13	0.887	36.53	0.879	14.8
TSDN_CSL2	38.69	0.914	37.53	0.894	36.78	0.885	14.8
TSDN_ESL1	38.54	0.910	37.00	0.886	36.32	0.877	3.8
TSDN_ESL2	38.73	0.912	37.27	0.889	36.57	0.880	3.8
TSDN_ESL3	38.75	0.914	37.37	0.892	36.69	0.883	3.8
TSDN_ESL4	38.23	0.905	36.52	0.879	35.69	0.866	3.8
TSDN_ESL5	38.37	0.908	36.74	0.882	35.94	0.871	3.8

The following settings are related to the ESL procedures. **TSDN_CSL1**: halve all channels (except for input and output channels of the network) of the expanded network at once, use **INIT1** described in III-C. **TSDN_CSL2**: reduce the channels step by step (illustrated in Figure 3), use **INIT1** described in III-C. **TSDN_ESL1**: remove the middle 4 layers of TSDN_CSL1 at once, and use **INIT2**. **TSDN_ESL2**: remove the middle 4 layers of TSDN_CSL2 at once, and use **INIT2**. **TSDN_ESL3**: remove the middle 2 layers of TSDN_CSL2, then remove the other 2 layers, use **INIT2**. **TSDN_ESL4**: the network architecture is the same as TSDN_ESL3 (tiny TSDN before expanding), but is trained from scratch. **TSDN_ESL5**: the network architecture is the same as TSDN_ESL3 (tiny TSDN before expanding), but is trained directly using **INIT1** and **INIT2**.

TABLE V

ABLATION STUDY OF ESL BEING APPLIED TO OTHER NETWORKS AND COMPARISON WITH KNOWLEDGE DISTILLATION METHOD

Models	×100		×250		×300		
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	Model Size (MB)
U-Net [1]	38.56	0.906	36.40	0.870	35.56	0.854	29.6
CBDNet [19]	38.12	0.893	35.70	0.842	34.84	0.818	16.7
U-Net-ESL	38.57	0.913	37.40	0.890	36.72	0.881	29.6
SU-Net	37.92	0.898	36.08	0.863	35.37	0.849	7.8
SU-Net-ESL	38.28	0.908	36.98	0.883	36.21	0.872	7.8
SU-Net-CD	38.23	0.904	36.46	0.873	35.73	0.860	7.8
TSDN_CSL2	38.69	0.914	37.53	0.894	36.78	0.885	14.8
TSDN_CSL3	38.46	0.907	36.74	0.883	35.96	0.871	14.8
TSDN	38.75	0.914	37.37	0.892	36.69	0.883	3.8

U-Net-ESL: train U-Net [1] using the proposed ESL method. **SU-Net**: train "small" version U-Net (all channels except input and output channels of U-Net [1] are halved) from scratch. **SU-Net-ESL**: train "small" version U-Net with the proposed ESL method. **SU-Net-CD**: use knowledge distillation algorithm CD [51] to train and compress U-Net [1] to "small" version U-Net. **TSDN_CSL2**: TSDN with similar model size as other models (about half of U-Net [1] and similar to CBDNet [19]) and is trained with ESL. **TSDN_CSL3**: TSDN with similar model size as other models and is trained from scratch.

we conducted ablation studies in which 3 initialization forms are employed: (1) First: network k+1 uses the first No and Ni channels' parameters of network k for initialization; (2) Last: network k+1 uses the last No and Ni channels' parameters of network k for initialization; (3) Combined: network k+1 uses the combination (average) of all channels' parameters of network k for initialization. As shown in Table III, the performance of different initialization strategies is similar. The average PSNR difference between the 3 strategies does not exceed 0.05 and the average SSIM difference is less than 0.02, which means the above strategies are basically equivalent.

TABLE VI
ABLATION STUDY OF THE MEAN AND STANDARD DEVIATION OF THE NOISE

	input	mid	output
abs_mean	476.03	42.37	41.80
std_dev	1321.21	246.60	238.28

In the first row, input, mid, and output denote the input image, output image of the first stage (TSDN_D), and the output image of TSDN, respectively.The absolute mean and standard deviation of noise are denoted as a and b, respectively. The results are average values of the Sony SID test set.

3) ESL: The ESL method is proposed to improve the performance of a tiny network, and the learning procedure of ESL is with **INIT1** and **INIT2** in a fine-grained way. Table IV demonstrates the effectiveness of ESL. First of all, if the small network is trained directly on the dataset (TSDN_ESL4) the performance is not satisfactory, while the ESL method (TSDN_ESL4) helps the network improve its performance. Especially, when the noise is severe ($\times$300) the ESL helps the network increase the PSNR by 1, which is a large improvement. Second, the fine-grained learning process in the Channel-Shrink-Learning (CSL) and Layer-Shrink-Learning (LSL) steps also help improve performance. In CSL, as demonstrated in TSDN_CSL1 and TSDN_CSL2 in Table IV, the step-by-step strategy helps the network learn better from the large network. And as demonstrated in TSDN_ESL2 and TSDN_ESL3 in the table, modifying the network with finer granularity in LSL helps improve performance. Besides, **INIT1** and **INIT2** are helpful for performance (see TSDN_ESL4 and TSDN_ESL5). In conclusion, both the initialization method **INIT1** and **INIT2** and the fine-grained learning procedures are helpful for performance.

To further exploit the effectiveness and generalizability of the proposed ESL method, we apply ESL to other networks, including U-Net [1] and "small" U-Net (SU-Net, all channels of U-Net are halved except input and output channels). For U-Net [1], we double its internal channels to get the expanded network and then use the ESL method to train and shrink it to the original U-Net. And for the "small" U-Net (SU-Net), we use the U-Net [1] as the expanded network and train it with ESL. As shown in Table V, compared to training from scratch (U-Net [1] and SU-Net), the proposed ESL method can improve the performance of both U-Net [1] and "small" U-Net (U-Net-ESL and SU-Net-ESL). The performance of U-Net-ESL is similar to the proposed TSDN while the model size of U-Net is 8 times that of TSDN.

Besides, we conduct experiments on TSDN with and without ESL when its model size is similar to CBDNet [19] (half of U-Net [1]). As shown in Table V, when trained with ESL, the performance of TSDN (TSDN_CSL2) is better than U-Net [1], CBD-Net [19] and U-Net-ESL. And when trained from scratch, the performance of TSDN (TSDN_CSL3) is better than CBD-Net [19]. Compared to U-Net [1], the average PSNR and SSIM of TSDN_CSL3 is 0.21 and 0.010 higher than U-Net [1], respectively. Besides, in darker environments ($\times$250 and $\times$300), TSDN_CSL3 outperforms U-Net [1] more.

Moreover, we perform experiments to compare the performance of the ESL method and the knowledge distillation method. We choose a representative knowledge distillation algorithm for low-level vision tasks, CD [51], to compress U-Net [1]. The original U-Net [1] is used as the teacher network and the "small" U-Net (SU-Net-CD) is the student network. Note that SU-Net-ESL and SU-Net-CD have the same architecture and model size as "small" U-Net (SU-Net). As shown in SU-Net and SU-Net-CD in Table V, the knowledge distillation method CD [51] can improve the performance of SU-Net. Nevertheless, the performance of the proposed ESL method is superior to CD [51] (SU-Net-ESL compared to SU-Net-CD).

4) Noise Distribution: In section III-A, we have analyzed the noise model in extremely low-light conditions. The noise distribution analyzed in III-A is based on a certain pixel located on x and the mean and standard deviation are its statistical properties, which can be used to characterize noise levels. When the noise is severe, the major parts in Equation 2 are $\eta_g(x)$ and $\xi(x)$, which are independent with signal. Thus the statistical distribution of noise at different locations can be used to measure the noise distribution at certain locations. Besides, assume that the label images of the dataset are "clean" images and then the noise can be calculated by subtracting the "clean" image from the noisy one. As demonstrated in Table VI, after the first stage (TSDN_D) processing, the absolute mean and standard deviation of the noise are reduced by a large amount, which means most of the noise is removed in the first stage. And then absolute mean and standard deviation are further reduced in the second stage (TSDN_R), which is responsible for image restoration (the visual effect is in Figure 1). In our work, we use the clean images in the dataset to constrain the intermediate results, which makes the training convenient and does not require additional datasets. This constraint may not produce the most suitable intermediate results for the network to learn and we will further research to find out the best intermediate constraint for TSDN.

V. CONCLUSION

In this paper, to address the low light denoising problem, denoising in the dark is modeled as noise removal and image restoration and a novel algorithm Two-Stage-Denoising (TSDN) is proposed. In TSDN, the noise-removal stage is responsible for removing the noise and then an intermediate image is obtained. Then, in the restoration stage, a clean image is recovered from the intermediate image. To train the tiny TSDN, we present an Expand-Shrink-Learning (ESL) method that leverages the learning capability of a large network to improve the performance of the tiny network. Experimental results show that the TSDN achieves SOTA performance in the dark environment with a smaller model size. TSDN achieves 1 higher PSNR than U-Net for denoising [1] in extremely low-light conditions ($\times$250 and $\times$300 in Table I) while the model size is one-eighth of U-Net. Besides, the ESL method can help the network improve PSNR by 1 when the light is extremely low.

REFERENCES

[1] C. Chen, Q. Chen, J. Xu, and V. Koltun, "Learning to see in the dark," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3291–3300.

[2] K. Wei, Y. Fu, Y. Zheng, and J. Yang, "Physics-based noise modeling for extreme low-light photography," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 8520–8537, Nov. 2022.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[4] A. Abdelhamed, M. Brubaker, and M. Brown, "Noise flow: Noise modeling with conditional normalizing flows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3165–3173.

[5] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2005, pp. 60–65.

[6] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[7] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian, "Practical poissonian-Gaussian noise modeling and fitting for single-image raw-data," *IEEE Trans. Image Process.*, vol. 17, no. 10, pp. 1737–1754, Oct. 2008.

[8] U. Schmidt and S. Roth, "Shrinkage fields for effective image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2774–2781.

[9] Y. Chen, W. Yu, and T. Pock, "On learning optimized reaction diffusion processes for effective image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5261–5269.

[10] X. Mao, C. Shen, and Y.-B. Yang, "Image restoration using very deep convolutional encoder–decoder networks with symmetric skip connections," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 1–9.

[11] 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.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.

[12] C. Chen, Z. Xiong, X. Tian, and F. Wu, "Deep boosting for image denoising," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 3–18.

[13] S. Cheng, Y. Wang, H. Huang, D. Liu, H. Fan, and S. Liu, "NBNet: Noise basis learning for image denoising with subspace projection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 4894–4904.

[14] A. Abdelhamed, S. Lin, and M. S. Brown, "A high-quality denoising dataset for smartphone cameras," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1692–1700.

[15] T. Plötz and S. Roth, "Benchmarking denoising algorithms with real photographs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2750–2759.

[16] X. Li, "Demosaicing by successive approximation," *IEEE Trans. Image Process.*, vol. 14, no. 3, pp. 370–379, Mar. 2005.

[17] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, 2015, pp. 234–241.

[18] K. Wei, Y. Fu, J. Yang, and H. Huang, "A physics-based noise formation model for extreme low-light raw denoising," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2755–2764.

[19] S. Guo, Z. Yan, K. Zhang, W. Zuo, and L. Zhang, "Toward convolutional blind denoising of real photographs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1712–1722.

[20] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–9.

[21] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," 2015, *arXiv:1510.00149*.

[22] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. Peter Graf, "Pruning filters for efficient ConvNets," 2016, *arXiv:1608.08710*.

[23] H. Wang, X. Hu, Q. Zhang, Y. Wang, L. Yu, and H. Hu, "Structured pruning for efficient convolutional neural networks via incremental regularization," *IEEE J. Sel. Topics Signal Process.*, vol. 14, no. 4, pp. 775–788, May 2020.

[24] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1," 2016, *arXiv:1602.02830*.

[25] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 6869–6898, 2017.

[26] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 525–542.

[27] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, "DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients," 2016, *arXiv:1606.06160*.

[28] M. J. McDonnell, "Box-filtering techniques," *Comput. Graph. Image Process.*, vol. 17, no. 1, pp. 65–70, Sep. 1981.

[29] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. 6th Int. Conf. Comput. Vis.*, 1998, pp. 839–846.

[30] C. Ren, X. He, C. Wang, and Z. Zhao, "Adaptive consistency prior based deep network for image denoising," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 8592–8602.

[31] Z. Zha, B. Wen, X. Yuan, S. Ravishankar, J. Zhou, and C. Zhu, "Learning nonlocal sparse and low-rank models for image compressive sensing: Nonlocal sparse and low-rank modeling," *IEEE Signal Process. Mag.*, vol. 40, no. 1, pp. 32–44, Jan. 2023.

[32] Z. Zha, X. Yuan, B. Wen, J. Zhou, J. Zhang, and C. Zhu, "From rank estimation to rank approximation: Rank residual constraint for image restoration," *IEEE Trans. Image Process.*, vol. 29, pp. 3254–3269, 2020.

[33] Z. Zha, X. Yuan, J. Zhou, C. Zhu, and B. Wen, "Image restoration via simultaneous nonlocal self-similarity priors," *IEEE Trans. Image Process.*, vol. 29, pp. 8561–8576, 2020.

[34] Z. Zha, X. Yuan, B. Wen, J. Zhang, J. Zhou, and C. Zhu, "Image restoration using joint patch-group-based sparse representation," *IEEE Trans. Image Process.*, vol. 29, pp. 7735–7750, 2020.

[35] Z. Zha, X. Yuan, B. Wen, J. Zhou, J. Zhang, and C. Zhu, "A benchmark for sparse coding: When group sparsity meets rank minimization," *IEEE Trans. Image Process.*, vol. 29, pp. 5094–5109, 2020.

[36] Z. Zha, X. Yuan, B. Wen, J. Zhou, and C. Zhu, "Group sparsity residual constraint with non-local priors for image restoration," *IEEE Trans. Image Process.*, vol. 29, pp. 8960–8975, 2020.

[37] E. Schwartz, R. Giryes, and A. M. Bronstein, "DeepISP: Toward learning an end-to-end image processing pipeline," *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 912–923, Feb. 2019.

[38] Y. Lu and S. Jung, "Progressive joint low-light enhancement and noise removal for raw images," *IEEE Trans. Image Process.*, vol. 31, pp. 2390–2404, 2022.

[39] M. Lamba and K. Mitra, "Restoring extremely dark images in real time," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 3486–3496.

[40] C. Chen, Q. Chen, M. Do, and V. Koltun, "Seeing motion in the dark," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3184–3193.

[41] H. Jiang and Y. Zheng, "Learning to see moving objects in the dark," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 7323–7332.

[42] S. W. Zamir et al., "CycleISP: Real image restoration via improved data synthesis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2693–2702.

[43] W. Wang, X. Chen, C. Yang, X. Li, X. Hu, and T. Yue, "Enhancing low light videos by exploring high sensitivity camera noise," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4110–4118.

[44] C. Bucilua, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2006, pp. 535–541.

[45] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[46] A. Romero, N. Ballas, S. Ebrahimi Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for thin deep nets," 2014, *arXiv:1412.6550*.

[47] S. You, C. Xu, C. Xu, and D. Tao, "Learning from multiple teacher networks," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 1285–1294.

[48] Y. Wang, C. Xu, C. Xu, and D. Tao, "Adversarial learning of portable student networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, 2018, pp. 1–8.

[49] S. Guo, J. M. Alvarez, and M. Salzmann, "ExpandNets: Linear over-parameterization to train compact convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1298–1310.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–15.

[51] H. Wang, Y. Li, Y. Wang, H. Hu, and M. Yang, "Collaborative distillation for ultra-resolution universal style transfer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1857–1866.

Wenshu Chen received the B.Eng. degree in microelectronics from Fudan University, Shanghai, China, in 2021, where he is currently pursuing the Ph.D. degree (a Bachelor-Straight-to-Doctorate Student) with the State Key Laboratory of ASIC and System, School of Microelectronics. His current research interests include image processing, computer vision, and domain-specific processor design.

Yujie Huang received the B.Eng. degree in microelectronics from Xidian University, Xi'an, China, in 2016, and the M.Sc. degree in microelectronics from Fudan University, Shanghai, China, in 2019, where he is currently pursuing the Ph.D. degree with the State Key Laboratory of ASIC and System, School of Microelectronics. His current research interests include computer vision, image processing, IC design, and artificial intelligence.

Mingyu Wang (Member, IEEE) received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, in 1999, the M.S. degree in electronic engineering from Shanghai Jiao Tong University, China, in 2001, and the Ph.D. degree from the School of Microelectronics, Fudan University, Shanghai, China, in 2011. He is currently an Associate Professor with the State Key Laboratory of ASIC and System, Fudan University. His research interests include artificial intelligence chip design, mixed signal circuit design, and low-power SOC design.

Xiaolin Wu (Fellow, IEEE) received the B.Sc. degree from Wuhan University in 1982 and the Ph.D. degree from the University of Calgary in 1988. He started his academic career in 1988. He has been a Faculty Member with Western University and New York Polytechnic University (NYU Poly). He is currently with McMaster University, where he holds an NSERC Senior Industrial Research Chair. He is also a McMaster Distinguished Engineering Professor. His research interests include image processing, computer vision, and multimedia signal coding and communication. He has published over 260 research articles and holds five patents in these fields. He serves on the IEEE Industrial Digital Signal Processing Committee, the IEEE Multidimensional Signal Processing Committee, and on the technical committees for many IEEE international conferences/workshops. He is an Associate Editor of IEEE TRANSACTIONS ON IMAGE PROCESSING.

Xiaoyang Zeng (Member, IEEE) received the Ph.D. degree from the Chinese Academy of Sciences in 2001. He is currently the Executive Director of the State Key Laboratory of ASIC and System and the Vice Dean of the School of Microelectronics, Fudan University. His research interests include high-performance and low-power VLSI architecture design for information security algorithms, digital signal processing algorithms, wireless communication base-band processing, and mixed-signal circuits design technology. He also serves as the Steering Committee Member for ASP-DAC and a TPC Member for A-SSCC. He also serves as the Co-Chair for the Circuit and System Division, Chinese Institute of Electronics, and the TPC Chair for ASICON 2009/2013.