IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 9, 2021, accepted February 15, 2021, date of publication February 22, 2021, date of current version March 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3061062

A Residual Dense U-Net Neural Network for Image Denoising

JAVIER GURROLA-RAMOS[1], OSCAR DALMAU[1], AND TERESA E. ALARCÓN[2]
[1]Department of Computer Science, Mathematics Research Center, Guanajuato 36023, Mexico
[2]Department of Computer Science and Engineering, Centro Universitario de los Valles, Ameca 46600, Mexico

Corresponding author: Oscar Dalmau (dalmau@cimat.mx)

This work was supported in part by the Consejo Nacional de Ciencia y Tecnología (CONACYT), Mexico, under Grant 258033, and in part by the Laboratorio de Supercómputo del Bajío Project under Grant 300832.

ABSTRACT In recent years, convolutional neural networks have achieved considerable success in different computer vision tasks, including image denoising. In this work, we present a residual dense neural network (RDUNet) for image denoising based on the densely connected hierarchical network. The encoding and decoding layers of the RDUNet consist of densely connected convolutional layers to reuse the feature maps and local residual learning to avoid the vanishing gradient problem and speed up the learning process. Moreover, global residual learning is adopted such that, instead of directly predicting the denoised image, the model predicts the residual noise of the corrupted image. The algorithm was trained for the case of additive white Gaussian noise and using a wide range of noise levels. Hence, one advantage of the proposal is that the denoising process does not require prior knowledge about the noise level. In order to evaluate the model, we conducted several experiments with natural image databases available online, achieving competitive results compared with state-of-the-art networks for image denoising. For comparison purpose, we use additive Gaussian noise with levels 10, 30, 50. In the case of grayscale images, we achieved PSNR of 34.39, 29.11, 26.99, and SSIM of 0.9297, 0.8193, 0.7491. For color images we obtained PSNR of 36.68, 31.43, 29.12, and SSIM of 0.9600, 0.8961, 0.8465.

INDEX TERMS Additive white Gaussian noise, convolutional neural networks, image denoising, residual dense neural network.

I. INTRODUCTION

Image denoising is an important problem in the area of low-level image processing. The main objective of image denoising problem is to recover a clean image $\boldsymbol{x}$, which has been corrupted by some noise $\boldsymbol{v}$ from a source. One assumption in this work is that $\boldsymbol{v}$ is additive white Gaussian noise (AWGN); therefore, the noisy image $\boldsymbol{y}$ follows the degradation model $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{v}$.

Traditional model-based methods such as non-local means (NLM) [1], block-batching and 3-D filtering (BM3D) [2], weighted nuclear norm minimization (WNNM) [3] rely on image prior modeling, and their optimization algorithms are time-consuming. In recent years, we have witnessed a dramatic upsurge of exploiting convolutional neural networks (CNNs) toward solving image denoising [4].

Compared to traditional model-based methods, CNNs offer fast inference and good performance. In recent years, the complexity of CNN architectures has increased along with the increase in the number of parameters. At the same time, many of these models have also improved their performance [5], [6]. On the other hand, several models need prior knowledge about the type and level of noise or an estimation thereof to obtain their best performance [7]. Some recent neural networks for image denoising, with fewer parameters, have been proposed to achieve competitive results [8]–[10]. Nevertheless, these models are trained for a specific noise level, requiring a model instance for every noise level.

On the other hand, the U-Net model [11] was proposed for the task of semantic segmentation. The U-Net model's architecture consists of a contracting (encoder) path to capture context and a symmetric expanding path (decoder) to estimate the segmentation. This architecture has also recently been used for image denoising in models such as multi-level wavelet CNN (MWCNN) [12], densely connected

The associate editor coordinating the review of this manuscript and approving it for publication was Mauro Gaggero.

hierarchical image denoising network (DHDN) [5], and deep iterative down-up CNN (DIDN) [6] obtaining good results.

In this paper, we present a residual dense neural network (RDUNet) for image denoising with competitive results with state of the art. Its design is based on the DHDN architecture that combines densely connected convolutional blocks and allows the reuse of feature maps within the encoding and decoding sections and between themselves. Although the number of parameters of the RDUNet is greater than conventional methods, the design of the proposed model allows a smaller number of multiply-accumulate operations to be carried out compared to models with similar or even higher complexity.

The main contributions of our work are summarized as follows:

- We present a convolutional neural network based on DHDN model for natural image denoising, which is capable of handling a wide range of Gaussian noise levels.
- In the subsampling operation we use strided convolution, that allows us to obtain local contrast information, unlike max-pooling which measures homogeneity.

We train our model with images corrupted by additive white Gaussian noise with randomly selected levels. According to experiments, using color and gray level image datasets, the proposed model achieves competitive results compared to the state of the art and outperforms conventional methods.

The rest of this paper is organized as follows. In Section II, we provide a brief review of related works. In Section III, we introduce the proposed RDUNet model. In Section IV, we present the ablation study of the proposed model. In Section V, we report the experiment results of the proposed method and in Section VI we present the limitations of this study. Finally, in Section VII, we provide the conclusions of this paper.

II. RELATED WORK

There have been several approaches based on neural networks to handle the image denoising problem. In [13] Jain and Seung proposed the earliest convolutional neural network for natural image denoising. Burger *et al.* [14] proposed a multi-layer perceptron (MLP), the authors concluded that neural networks can achieve better results than traditional methods as BM3D [2] when considering the depth of the network, the size of the appropriate training patches, and training set. Zhang *et al.* [15] proposed a deep convolutional neural network for image denoising (DnCNN). This model improves the denoising performance by stacking multiple blocks of convolutional layers, batch normalization [16], rectified linear unit (ReLU) activations and the use of residual learning. In [17] Tai *et al.* proposed a deep end-to-end persistent memory network for image restoration, this model fuses both short-term and long-term memories to capture different levels of information. The fast and flexible denoising CNN (FFDNet) proposed by Zhang *et al.* [18] introduces the noise feature map for handle non-uniform noise level and downsampled sub-images for increasing the receptive field and the performance speed. The work in [19] proposes a generative adversarial neural network (GAN) to estimate the noise distribution and generate noise samples. Then, the noise patches sampled from the generator are used to build the training dataset. Kumwilaisak *et al.* [20] combines a multipath CNN and long-short term memory (LSTM) layers to denoise images corrupted by Poisson noise.

More recently models such as DHDN [5], DIDN [6], and residual dense network (RDN) [21] have improved baseline results established by DnCNN and FFDNet models. However, these models have significantly increased their complexity and the amount of parameters to achieve such improvement. Additionally, to boost the performance, several recent methods applies techniques as self-ensemble and model ensemble [22]. In self-ensemble method, the results from eight flipped/rotated versions of the same image are averaged together. In model ensemble method, the outputs of several models are averaged. In the latter case, it is necessary to train more than one model to apply this alternative.

Several models of the state of the art are based on network architectures such as autoencoders, residual, dense or a combination thereof. The U-Net model [11] is one of the most widely used autoencoder architectures today. The encoding section consists of several convolutions, and max-pooling that halves the size of the feature maps at each level, while doubling the number of feature maps. The decoding section restores the size of the feature maps and keeps a symmetric form with respect to the encoding section. This symmetry, enables the reuse of feature maps by concatenating features maps at the same level and reduces the loss of information caused by the encoding-decoding process. In [12], Liu *et al.* propose the MWCNN model, which combines wavelet transforms and convolutional layers inside of a U-Net model, instead of simple convolutions and max-poling. Wang *et al.* [23] extends the MWCNN adding residual dense blocks in each layer of the model, improving the performance of the base network. The DIDN model proposed by Yu *et al.* [6] uses several U-Net based blocks, one behind the other which decrease and increase the image size several times. In [5], Park *et al.* propose the DHDN, this model uses the U-Net architecture as a framework but interchanging the simple convolution for densely connected residual blocks and doubles the number of feature maps present at each level of the network.

Residual learning and residual neural networks (ResNets), on the other hand, were proposed by He *et al.* [24] for addressing the network degradation problem as the network depth increases. This learning scheme combines (adds) the extracted features and the input of a sequence of convolutional layers, which can solve the vanishing or exploding gradient problems. Dense networks [25] reuses each of the generated feature maps as input from subsequent convolutions within the same convolutional block. These model architectures and the residual learning became the state-of-the art methods for image classification. In recent

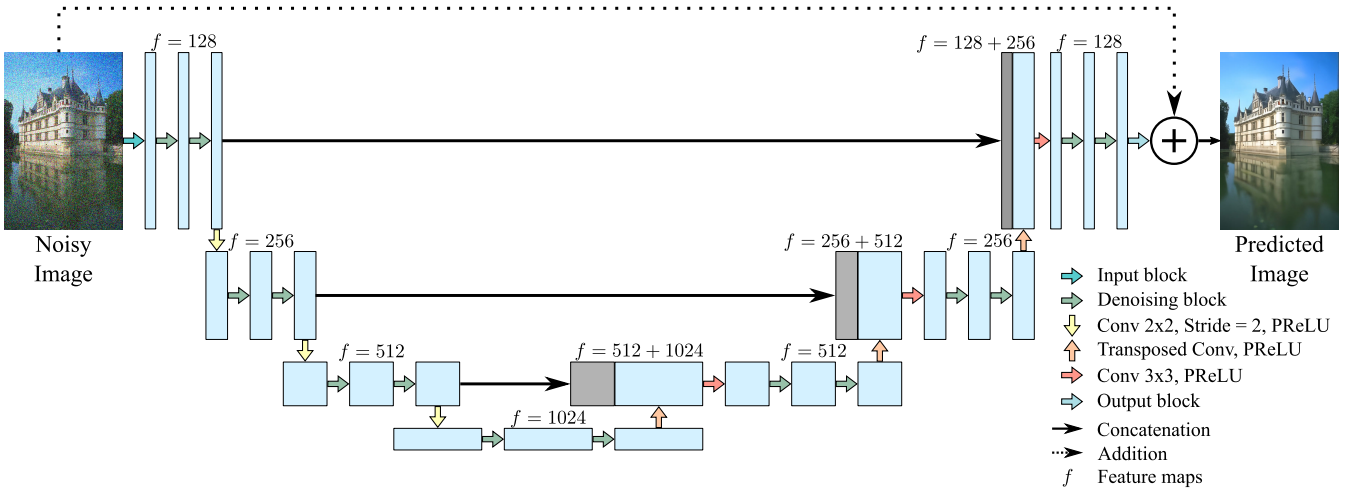


FIGURE 1. Architecture of proposed network.

years, several methods based on residual learning and dense connectivity have been proposed for image denoising. In [26], Rem *et al.* propose a deep residual network using depth-wise separable convolutions for denoising either for Gaussian or Poisson corrupted images. In [27], Zhang *et al.* propose a deep residual network with parametric rectifier linear unit layer for image restoration tasks: Gaussian image denoising, single image super resolution and JPEG image deblocking. In [21], Zhang *et al.* propose a model for Image super-resolution and image denoising that combines dense connectivity with residual learning. The residual dense blocks consist on several densely connected convolutions. A final convolution generates the same number of feature maps that the input of the block, along with a residual learning.

III. PROPOSED METHOD

The architecture of the proposed model is illustrated in Fig. 1. Residual learning is adopted for the proposed RDUNet model. Consider a clean image $\boldsymbol{x}$ corrupted by additive white Gaussian noise $\boldsymbol{v}$: $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{v}$. The problem of recovering $\boldsymbol{x}$ can be formulated as finding a parametric function $\mathcal{F}(\cdot; \boldsymbol{\Theta})$ such that

$$\hat{\boldsymbol{x}} = \mathcal{F}(\boldsymbol{y}; \boldsymbol{\Theta}) \tag{1}$$

provides an estimate $\hat{\boldsymbol{x}}$ of the original image $\boldsymbol{x}$ given the corrupted image $\boldsymbol{y}$ and parameters $\boldsymbol{\Theta}$. Due to the corrupted image $\boldsymbol{y}$ contains most of the structure of the clean image $\boldsymbol{x}$ it is reasonable to preserve this structure and estimate just the added noise. For this purpose, suppose we have a parametric mapping $\mathcal{H}(\cdot; \Theta)$ such that $\mathcal{H}(\boldsymbol{y}; \Theta) \approx -\boldsymbol{v}$. Therefore, the denoising parametric model based on residual learning can be written as follows:

$$\mathcal{F}(\boldsymbol{y}; \boldsymbol{\Theta}) = \mathcal{H}(\boldsymbol{y}; \boldsymbol{\Theta}) + \boldsymbol{y}, \tag{2}$$

where $\boldsymbol{\Theta}$ represents the set of trainable parameters of the model. Consequently, in order to estimate the parameters, we can solve the following optimization problem:

$$\boldsymbol{\Theta}^* = \arg\min_{\boldsymbol{\Theta}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(\mathcal{F}(\boldsymbol{y}_i; \boldsymbol{\Theta}), \boldsymbol{x}_i\right) + \frac{\lambda}{2} \|\boldsymbol{\Theta}\|^2, \tag{3}$$

where $\{(\boldsymbol{y}_i, \boldsymbol{x}_i)\}$ represents a training dataset, $\boldsymbol{x}_i$ is the clean image and $\boldsymbol{y}_i$ the corresponding degraded image by additive white Gaussian noise. The first term in (3) corresponds to the fidelity term and the second term is the regularization term. The hyperparameter $\lambda > 0$ controls the tradeoff between these two terms. The function $\mathcal{L}(\cdot, \cdot)$ is the loss function, and in our case the L_1-norm provided a better result, i.e., we use the following function:

$$\mathcal{L}(\boldsymbol{y}, \boldsymbol{x}) = ||\boldsymbol{y} - \boldsymbol{x}||_1. \tag{4}$$

On the other hand, the main structure of the mapping $\mathcal{F}(\cdot; \boldsymbol{\Theta})$ is a U-Net model that corresponds to the mapping $\mathcal{H}(\cdot; \boldsymbol{\Theta})$, see Fig. 2, that consists of three encoding/decoding levels, a bottleneck level, and a shortcut between all the encoding and decoding layers at the same level 1. The main differences with the standard U-Net is that we change the Convolution-Maxpool blocks, for more specialized denoising blocks together with the idea of using dense networks and the noise residual modeling, see details in Sections III-A and III-B.

At each level, there are two denoising blocks in the encoding and decoding stages. The output of every encoding level is downsampled by a factor of 2 using a convolution with kernel size 2×2 and a stride of size 2 instead of max-pooling. With each downsampling, the number of feature maps is doubled in order to reduce the loss of information from one level to another. The upsampling in the decoding stage is performed by transposed convolutions. The shortcuts between encoding layers and decoding layers are handled

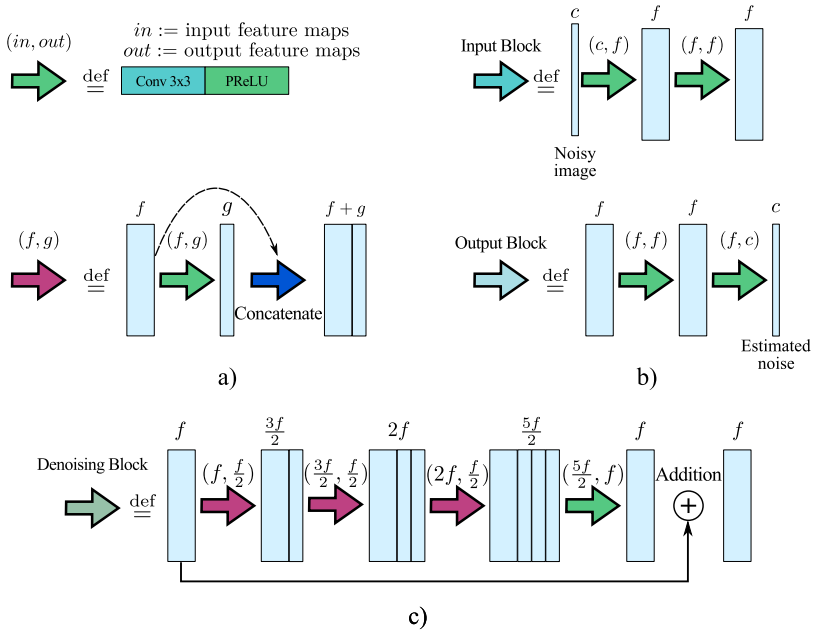


FIGURE 2. Building blocks of the RDUNet: a) Basic components used in the building blocks of the RDUNet model, b) input and output blocks, c) densely connected denoising block used throughout the model.

through concatenation. After each upsampling and concatenation, a 3×3 convolution is performed to reduce the number of features and smooth the upsampled features, keeping the most important information from these sources.

Throughout the model, the parametric ReLU (PReLU) activation function is adopted as shown in Fig. 2. When we use the PReLU function, there are as many trainable parameters as the number of feature maps in the corresponding layer. This activation function increases the model flexibility without introducing a large number of extra parameters [28].

The input block for feature extraction is shown in Fig. 2 (a). The spatial information can be reasonably useful for predicting a given pixel's actual value for the denoising task. Locally, the neighbor pixels have a similar value to the current pixel to predict. High noise levels usually require larger patch sizes to capture context information [29]. One way to get more spatial information is by selecting the kernel size larger than the typical 3×3 size. However, using this approach increases the number of parameters quadratically for the spatial dimension using a kernel of size $k \times k$.

For the RDUNet, we increase the field of view by staking two convolutions with kernel size 3×3 instead of using just one with a larger kernel size. Both convolutions generate 128 feature maps to capture as much information as possible from the corrupted image.

A. DENOISING BLOCK

The denoising block is based on the bottleneck block of ResNet50 [24] and on the reuse of feature maps from the DenseNet model [25], see Fig. 2 (b). First, we perform a 3×3 convolution for reducing the number of feature maps (f) in half. Afterward, two convolutions with kernel size 3×3 are performed. These convolutions use all the preceding feature maps as input information, including block input feature maps. Finally, a 3×3 convolution summarizes all the previous feature maps generated and the denoising block's input. This last convolution generates the same amount of feature maps as the denoising block's input to carry out the local residual learning. Unlike the block proposed for the RDN network, we use a 3×3 kernel size convolution instead of 1×1 to consider spatial information by merging previously generated feature maps at the cost of increasing the number of parameters. As mentioned above, the number of parameters in the denoising block increases quadratically with respect to the size of the convolution kernel and is also proportional to the number of feature maps.

B. OUTPUT BLOCK

The output block is shown in Fig. 2 (c). This block reduces the number of feature maps to match the input noisy image's size and generates an estimate of the residual noise. Note that the output block is very similar to the input block, with the difference that, in the output block we map an image of size $H \times W \times f$ to one of size $H \times W \times c$. This block's output is used for global residual learning, adding its result to the input image to finally obtain the denoised image.

IV. ABLATION STUDY

Table 1 shows the ablation study. In this Table, we present the effects of the denoising block's design, the subsampling method, and the use of global residual learning. The variants of the proposed model are denoted in the form RDUNet_{vj} where $j = 0, \ldots, 6$, and the proposed model is just denoted

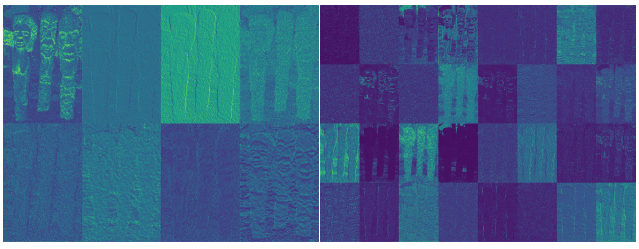
TABLE 1. Ablation study of the main components of the proposed model: denoising block with 1 × 1 bottleneck convolution ($DB_{1\times1}$), denoising block with 3 × 3 bottleneck convolution ($DB_{3\times3}$), max pooling with kernel size 2 × 2 ($MP_{2\times2}$), convolution with kernel size 2 × 2 with stride = 2 as subsampling method ($Conv_{2\times2}$) and global residual learning (GRL). We also compare the proposed RDUNet model with DHDN and U-Net models, and consider some of their main differences: Input/Output convolutions with 1 × 1 kernel size ($I/O_{1\times1}$) or with 3 × 3 kernel size ($I/O_{3\times3}$), upsampling by pixel shuffle (PS) or by transposed convolution ($Conv^T$), additional shortcut in the bottleneck (BN-shc) and the number of levels of the model. We also report the number of trainable parameters, multiply-accumulate (MAC) operations on 64 × 64 color images, and the PSNR value on CBSD68 color dataset with noise level $\sigma = 50$.

Models	Additional building blocks						Denoising block		Subsampling			Complexity and performance		
	Levels	$I/0_{1\times1}$	$I/0_{3\times3}$	BN-shc	PS	$Conv^T$	$DB_{1\times1}$	$DB_{3\times3}$	$MP_{2\times2}$	$Conv_{2\times2}$	GRL	#Param.	MACs	PSNR (dB)
U-Net	5	✓	✗	✗	✗	✓	✗	✗	✓	✗	✗	31M	15G	27.35
DHDN	4	✓	✗	✓	✓	✗	✗	✗	✓	✗	✓	168M	64G	27.71
$RDUNet_{v0}$	4	✗	✓	✗	✗	✓	✓	✗	✓	✗	✗	84M	28G	28.27
$RDUNet_{v1}$	4	✗	✓	✗	✗	✓	✓	✗	✓	✗	✓	84M	28G	28.28
$RDUNet_{v2}$	4	✗	✓	✗	✗	✓	✓	✗	✗	✓	✗	97M	29G	28.29
$RDUNet_{v3}$	4	✗	✓	✗	✗	✓	✓	✗	✗	✓	✓	97M	29G	28.29
$RDUNet_{v4}$	4	✗	✓	✗	✗	✓	✗	✓	✓	✗	✗	148M	45G	28.29
$RDUNet_{v5}$	4	✗	✓	✗	✗	✓	✗	✓	✓	✗	✓	148M	45G	28.31
$RDUNet_{v6}$	4	✗	✓	✗	✗	✓	✗	✓	✗	✓	✗	166M	48G	28.34
RDUNet	4	✗	✓	✗	✗	✓	✗	✓	✗	✓	✓	166M	48G	28.38

as RDUNet. We also include a comparison with the U-Net and DHDN models.

We consider two versions of the convolution bottleneck. These versions only consider the last layer that maps from $5f/2$ to f feature maps in the denoising block, Fig. 2 (c). In the first version we use a convolution with kernel size 1×1, denoted as $DB_{1\times1}$, and in the second one, the kernel size is 3×3, denoted as $DB_{3\times3}$. Concerning the subsampling method, we consider max pooling with kernel size 2×2 and strided convolution, with kernel size 2×2 and stride size of 2, denoted as $MP_{2\times2}$ and $Conv_{2\times2}$ respectively. Besides, we study a global residual learning, denoted as GRL, to estimate the noise and afterward remove it from the input image.

As shown in Table 1, the number of parameters and multiplication-addition operations increase significantly when we use the denoising block $DB_{3\times3}$ instead of $DB_{1\times1}$. This is due to the denoising block is used twice for each level of the model, in both the encoder and decoder blocks. Similarly, when we use $Conv_{2\times2}$ subsampling method instead of $MP_{2\times2}$ the number of parameters increases. However, the number of multiplication-addition operations increases to a lesser extent compared to the previous case. Note that as the complexity of the model increases, so does its computational cost and performance. The learned $Conv_{2\times2}$ subsampling parameters can measure both contrast and homogeneity, the local happens occurs when the entries of the convolutional kernel combine positive and negative numbers, see Fig. 3. Unlike max-pooling, which measures homogeneity and only considers information within each feature map independently.



a) b)

FIGURE 3. Randomly selected feature maps: a) after applying the second denoising block and before applying the subsampling $Conv_{2\times2}$, b) after applying the subsampling $Conv_{2\times2}$.

V. EXPERIMENTS

A. EXPERIMENTAL SETTING FOR TRAINING

We use the DIV2K dataset [30] originally proposed for image super-resolution. The DIV2K training dataset consists of 800 high-quality, high-resolution color images (from size 648×2040 to 2040×2040). The DIV2K validation dataset consists of 100 images with similar characteristics to the training dataset.

For training the proposed model, we split the original DIV2K training dataset in input/output patches $\{(\boldsymbol{y}_i, \boldsymbol{x}_i)\}$ of size 64×64. We train a model for the case of color images and another for the case of grayscale images. In the case of the model for grayscale image, we first converted the color images into grayscale images. Noisy patches $\boldsymbol{y}_i$ are generated by corrupting the clean patches $\boldsymbol{x}_i$ with additive white Gaussian noise with noise intensity in the range $\sigma \in [5, 50]$. Additionally, we apply augmentation that consists on random vertical and horizontal flips, and 90° rotations. For the training, the algorithm only receives the pairs $\{(\boldsymbol{y}_i, \boldsymbol{x}_i)\}$; however, we don not provide information about the level of the noise.

In order to optimize eq. (3) we use the AdamW algorithm [31]. This optimization algorithm allows us to work the first and second terms of the loss function (3) decoupled. The regularization parameter used in eq. (4) is

TABLE 2. Performance of different denoising methods on BSD68, Kodak24 and Set12 grayscale datasets with different noise levels. The best two results of PSNR (dB) and SSIM are highlighted in red and blue respectively.

Methods	BSD68						Kodak24						Set12					
	$\sigma = 10$		$\sigma = 30$		$\sigma = 50$		$\sigma = 10$		$\sigma = 30$		$\sigma = 50$		$\sigma = 10$		$\sigma = 30$		$\sigma = 50$	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BM3D	33.32	0.9158	27.75	0.7731	25.60	0.6858	34.39	0.9127	29.12	0.7877	26.98	0.7140	34.40	0.9244	29.07	0.8305	26.36	0.7598
U-Net	33.47	0.9224	28.16	0.7948	26.04	0.7152	34.44	0.9182	29.30	0.8047	27.16	0.7354	34.21	0.9214	29.16	0.8369	26.85	0.7781
DnCNN	33.88	0.9270	28.36	0.7999	26.23	0.7189	34.90	0.9223	29.62	0.8071	27.49	0.7368	34.78	0.9270	29.52	0.8420	27.18	0.7826
IRCNN	33.74	0.9262	28.26	0.7989	26.19	0.7171	34.76	0.9215	29.52	0.8056	27.45	0.7342	34.72	0.9272	29.45	0.8392	27.14	0.7804
FFDNet	33.76	0.9266	28.39	0.8032	26.29	0.7245	34.81	0.9226	29.69	0.8123	27.62	0.7437	34.65	0.9271	29.61	0.8465	27.32	0.7903
RDN	34.00	N/A	28.56	N/A	26.41	N/A	35.17	N/A	30.00	N/A	27.85	N/A	35.06	N/A	29.94	N/A	27.60	N/A
DHDN	33.42	0.9213	28.55	0.8110	26.44	0.7296	34.43	0.9153	29.93	0.8211	27.88	0.7528	34.96	0.9315	29.91	0.8539	27.58	0.7984
DIDN	33.98	0.9284	28.58	0.8075	26.47	0.7310	35.16	0.9263	30.04	0.8222	27.96	0.7562	35.02	0.9315	30.00	0.8556	27.71	0.8025
RDUNet	33.98	0.9297	28.59	0.8104	26.48	0.7347	35.12	0.9272	30.01	0.8230	27.96	0.7584	34.99	0.9317	29.97	0.8553	27.68	0.8018
U-Net+	33.57	0.9236	28.22	0.7966	26.10	0.7176	34.54	0.9194	29.36	0.8066	27.22	0.7381	34.32	0.9224	29.24	0.8388	26.92	0.7808
RDN+	34.01	N/A	28.58	N/A	26.43	N/A	35.19	N/A	30.02	N/A	27.88	N/A	35.08	N/A	29.97	N/A	27.64	N/A
DHDN+	33.50	0.9230	28.59	0.8120	26.47	0.7308	34.54	0.9174	30.00	0.8237	27.93	0.7546	35.01	0.9320	29.97	0.8550	27.66	0.8004
DIDN+	34.01	0.9286	28.61	0.8081	26.50	0.7320	35.20	0.9267	30.08	0.8230	28.01	0.7576	35.07	0.9318	30.05	0.8565	27.77	0.8041
RDUNet+	34.01	0.9300	28.62	0.8112	26.51	0.7358	35.16	0.9277	30.04	0.8240	28.00	0.7597	35.03	0.9322	30.01	0.8560	27.73	0.8033

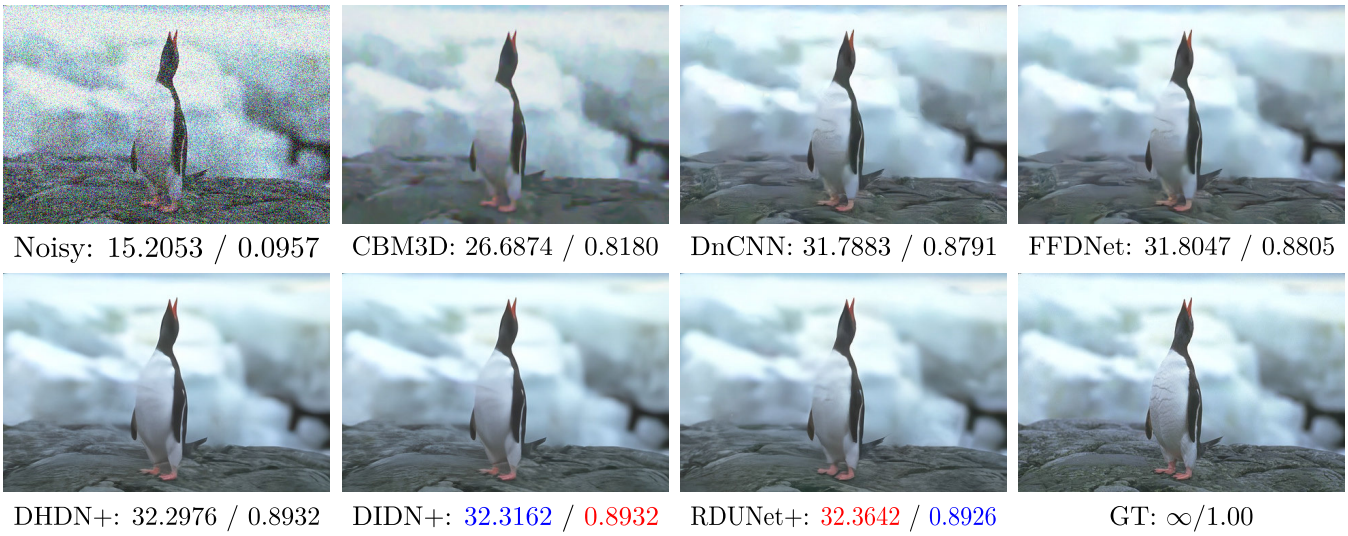


FIGURE 4. Comparison of denoising neural networks. *106024* image from CBSD68 dataset corrupted by AWGN with $\sigma = 50$. Their corresponding PSNR (dB)/SSIM values of denoised images are shown.

$\lambda = 10^{-5}$. The parameters of the AdamW algorithm are $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$. The initial learning rate is $\alpha_0 = 10^{-4}$, which is halved every 3 iterations over the complete dataset until it reaches a value of $\alpha_f = 10^{-6}$. The RDUNet model was trained with a batch size of 16 for 21 epochs. Our model was implemented in Python 3.6 using PyTorch framework. The training time was 48 hours in a Nvidia RTX Titan GPU. The source code and models are available at https://github.com/JavierGurrola/RDUNet.

B. TESTING, COMPARISONS AND RESULTS FOR AWGN

For evaluating the proposed RDUNet model, we use the CBSD68 [32], Kodak24 [33], and Urban100 [34] datasets for the case of color images. For grayscale images, we used the grayscale version of CBSD68 and Kodak24, and the Set12 dataset [15]. The CBSD68 dataset consists of 68 images of size 321×481. The Kodak24 dataset consists of 24 images of size 768×512. The Urban100 dataset is a collection of 100 high-resolution images with a variety of real-world structures. The Set12 dataset consists of seven images of size 256×256, and five images of size 512×512.

We compare our proposed RDUNet model with BM3D [2], [35], DnCNN [15], FFDNet [18], IRCNN [36], RDN [21], DHDN [5], DIDN [6]. In addition, we compare with the U-net model and we trained it from scratch. We also consider the self-ensemble [22] versions of the: U-Net, RDN,

FIGURE 5. Comparison of denoising neural networks. *189080* image from CBSD68 dataset corrupted by AWGN with $\sigma = 50$. Their corresponding PSNR (dB)/SSIM values of denoised images are shown.

DHDN, and DIDN models, denoted here as U-Net+, RDN+, DHDN+, and DIDN+ respectively. Similarly, we include the self-ensemble version, RDUNet+, of the proposed RDUNet.

For assessing the previous models, we use the peak signal-to-noise ratio (PSNR) average [37] and the structural similarity index (SSIM) average [38]. For comparison purposes, we report the results for noise levels $\sigma = \{10, 30, 50\}$, which are the most common used in the literature.

The results of the comparison appear in Table 2, for grayscale images, and in Table 3, for color images. In the case of the selected models for the comparison, most of the results are directly taken from the original paper. The U-net model was trained from scratch for grayscale and color images. The metric SSIM is not available for RDN and RDN+ models. In the case of the Urban100 and Set12 datasets, the DIDN, DHDN, DIDN+, and DHDN+ algorithms were evaluated with the pre-trained models available in their respective repositories. The rest of the methods were evaluated in all the datasets with their respective pre-trained models and their corresponding authors' source codes. In both tables, 2 and 3, the best and the second best results are highlighted in red and blue respectively. In Table 4, we summarize the number of times that a model achieves the best and second-best PSNR and SSIM values.

According to Table 4, the RDUNet+ achieves four times the best PSNR values, and six times the second-best values of the same metric, for both grayscale and color datasets. Note that even though the DIDN+ has more times best PSNR values, the gap between RDUNet+ and DIDN+ PSNR values, in the worst case, is lower than 0.09 dB.

In the case of SSIM metric, observe that the RDUNet+ model obtains six times the best values for grayscale and six times for color datasets, whereas the DIDN+ model obtains two times the best value for grayscale and two times for color dataset. Moreover, the RDUNet (without self-ensemble) outperforms most of the ensemble models, getting three and

FIGURE 6. Comparison of denoising neural networks. *227092* image from CBSD68 dataset corrupted by AWGN with $\sigma = 50$. Their corresponding PSNR (dB)/SSIM values of denoised images are shown.

FIGURE 7. Comparison of denoising neural networks. *kodim23* image from Kodak24 dataset corrupted by AWGN with $\sigma = 50$. Their corresponding PSNR (dB)/SSIM values of denoised images are shown.

five times the second-best value for the grayscale and color datasets respectively. Notice in Table 2 and Table 3 that, combining both metrics, the proposed model RDUNet+ appears more times in either first or second place than any other model.

Table 5 summarizes the results by grayscale and color datasets. The results were obtained using a weighted average, where the weights were calculated according to the size of the datasets. The RDUNet+ model achieves the best SSIM values in almost all cases, while the RDUNet obtains the

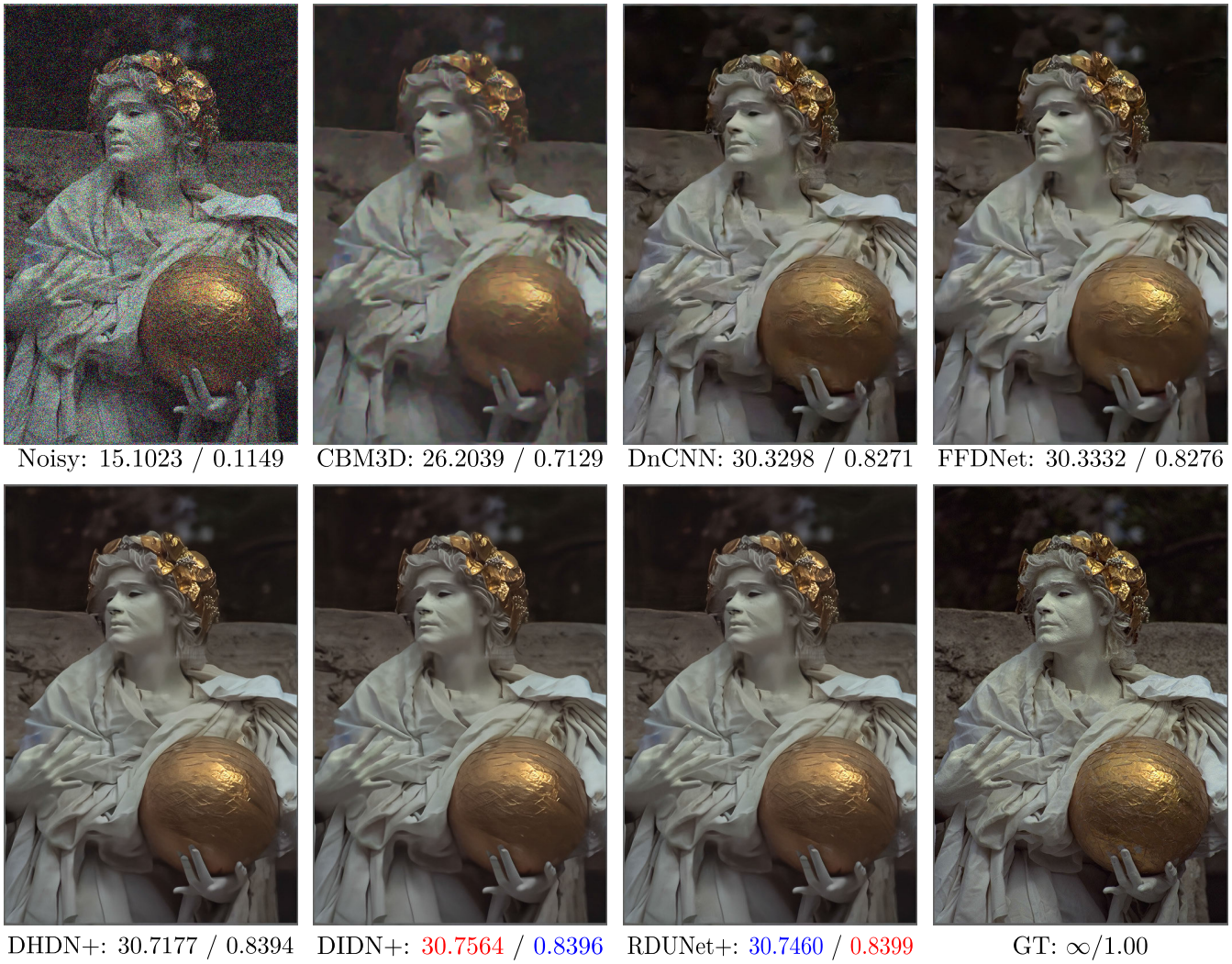


FIGURE 8. Comparison of denoising neural networks. *kodim17* image from Kodak24 dataset corrupted by AWGN with $\sigma = 50$. Their corresponding PSNR (dB)/SSIM values of denoised images are shown.

second-best value of the same metric in images with low noise level. On the other hand, the RDUNet+ achieves the best PSNR value in grayscale noisy images with $\sigma = 50$ and the second-best in grayscale images with noise levels $\sigma = 10$ and $\sigma = 30$, and color dataset with noise levels $\sigma = 30$ and $\sigma = 50$. In Table 6 we compile the number of times a method scored the best and second-best for both PSNR and SSIM metrics based on Table 5.

In order to visually compare the performance of some methods, from Fig. 4 to Fig. 10 we show results of denoised images, using the test datasets evaluated in this work. In the comparison, we consider the conventional method CBM3D, the DnCNN and FFDNet models, and the self-ensemble with best results: DHDN+, DIDN+ and RDUNet+. The later model are included because they obtained the best results in previous discussion.

In Fig. 4, the proposed model can recover more details of the penguin's legs, a better definition of the boundary between the rock and the background, and does not over smooth the penguin's chest. In Fig. 5, the RDUNet+ can recover more details in the forehead and the temple, and with fewer artifacts in the chin. However, models like DIDN+ and DHDN+ recover better the background of the image. In Fig. 6, the RDUNet+ model performances better in the upper half of the vase. Although, the model cannot satisfactorily recover the triangular pattern in the middle of the vase, it does retrieve the texture of that section. Fig. 7 shows that the performance of the DHDN+, DIDN+, and RDUNet+ is very similar, not finding significant visual differences. Nevertheless, the CBM3D, DnCNN, and FFDNet methods show artifacts in the background leaves due to the noise. Similar to the previous case, the performance of the DHDN+, DIDN+, and RDUNet+ in Fig. 8 is very similar, recovering part of the texture of the face of the statue but generating an artifact above the right hand, making that section blue. The CBM3D, DnCNN, and FFDNet methods show artifacts in the

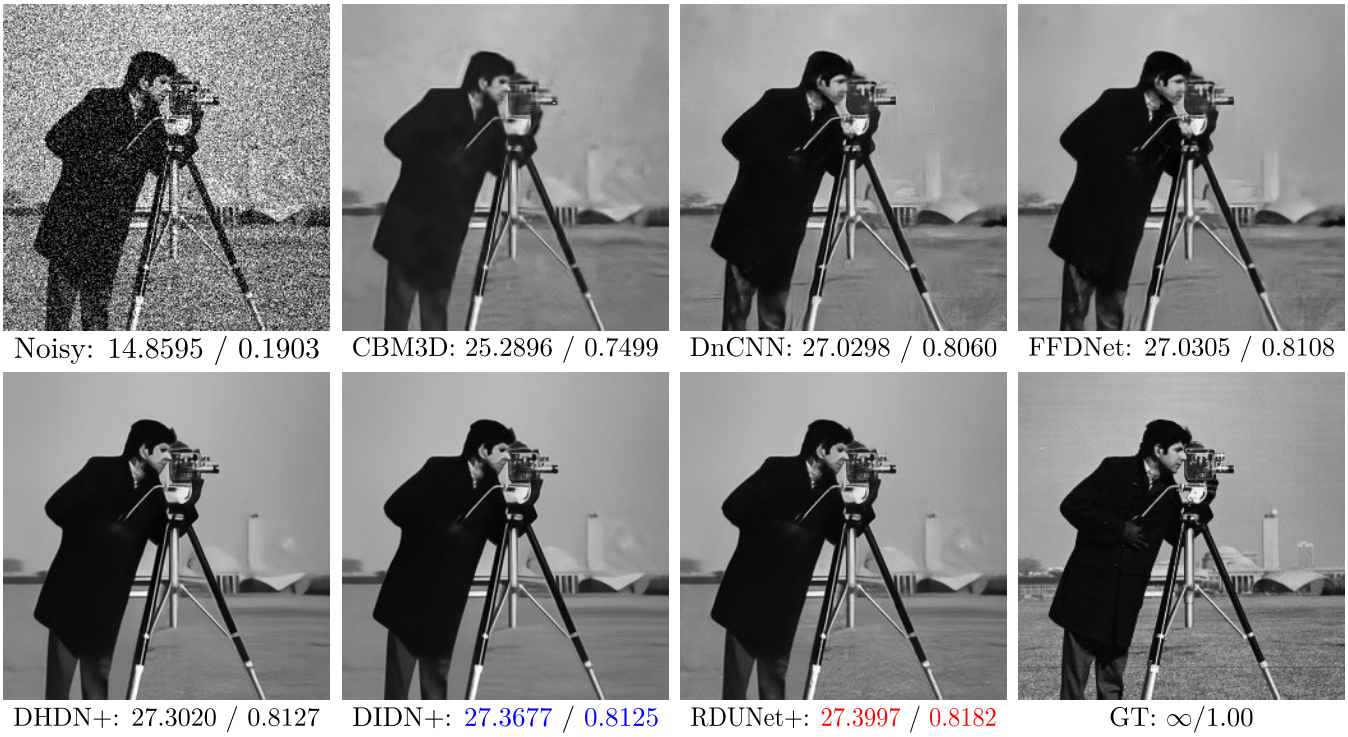


FIGURE 9. Comparison of denoising neural networks. *Camera man* image from Set12 dataset corrupted by AWGN with $\sigma = 50$. Their corresponding PSNR (dB)/SSIM values of denoised images are shown.

FIGURE 10. Comparison of denoising neural networks. *img_034* image from Urban100 dataset corrupted by AWGN with $\sigma = 50$. Their corresponding PSNR (dB)/SSIM values of denoised images are shown.

background; in particular, CBM3D changes the statue's color lightly. In Fig. 9, RDUNet+ recovers some texture from the grass and restores part of the background constructions. However, neither method is capable of recovering the left and right of the background buildings. Unlike the CBM3D, DnCNN, and FFDNet models, the DIDN+, DHDN+, and RDUNet models recover the sky better, but they smooth the cameraman's clothes. In Fig. 10, the bricks of the building are the elements that stand out the most. Observe that, the RDUNet+ and DIDN+ models are the methods that recover more details in the lower part of the building than in the upper part of it. These methods almost completely recover the clouds in this image.

Table 7 shows the number of multiplication-accumulation operations (MACs) and the number of parameters of DnCNN, FFDNet, U-Net, RDN, DIDN, DHDN, and RDUNet models.

TABLE 3. Performance of different denoising methods on CBSD68, Kodak24 and Urban100 color datasets with different noise levels. The best two results of PSNR (dB) and SSIM are highlighted in red and blue respectively.

Methods	CBSD68						Kodak24						Urban100					
	$\sigma = 10$		$\sigma = 30$		$\sigma = 50$		$\sigma = 10$		$\sigma = 30$		$\sigma = 50$		$\sigma = 10$		$\sigma = 30$		$\sigma = 50$	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
CBM3D	35.89	0.9512	29.71	0.8426	27.36	0.7632	36.57	0.9432	30.89	0.8459	28.62	0.7772	34.11	0.9307	27.94	0.8223	24.43	0.7256
U-Net	35.39	0.9484	29.74	0.8489	27.35	0.7711	35.89	0.9393	30.55	0.8446	28.11	0.7736	34.10	0.9461	29.03	0.8761	26.45	0.8134
DnCNN	36.12	0.9536	30.32	0.8611	27.92	0.7882	36.58	0.9446	31.28	0.8579	28.94	0.7915	36.21	0.9607	30.28	0.8922	28.16	0.8490
IRCNN	36.06	0.9533	30.22	0.8607	27.86	0.7889	36.70	0.9448	31.24	0.8581	28.92	0.7939	35.81	0.9581	30.28	0.8940	27.69	0.8396
FFDNet	36.14	0.9540	30.31	0.8603	27.96	0.7881	36.80	0.9462	31.39	0.8596	29.10	0.7949	35.77	0.9585	30.53	0.8983	28.05	0.8476
RDN	36.47	N/A	30.67	N/A	28.31	N/A	37.31	N/A	31.94	N/A	29.66	N/A	36.69	N/A	31.69	N/A	29.29	N/A
DHDN	36.05	0.9532	30.12	0.8579	27.71	0.7874	37.30	0.9509	31.98	0.8743	29.72	0.8170	36.45	0.9628	31.50	0.9135	29.15	0.8736
DIDN	36.48	0.9565	30.71	0.8706	28.35	0.8041	37.32	0.9500	31.97	0.8724	29.72	0.8156	36.55	0.9634	31.69	0.9161	29.38	0.8779
RDUNet	36.48	0.9571	30.72	0.8720	28.38	0.8067	37.29	0.9506	31.97	0.8738	29.72	0.8177	36.54	0.9636	31.63	0.9158	29.32	0.8774
U-Net+	35.64	0.9503	29.88	0.8525	27.48	0.7765	36.09	0.9410	30.68	0.8481	28.26	0.7795	34.42	0.9482	29.20	0.8797	26.59	0.8190
RDN+	36.49	N/A	30.70	N/A	28.34	N/A	37.33	N/A	31.98	N/A	29.70	N/A	36.75	N/A	31.78	N/A	29.38	N/A
DHDN+	36.27	0.9556	30.41	0.8654	28.02	0.7965	37.31	0.9510	31.99	0.8744	29.73	0.8170	36.55	0.9633	31.62	0.9149	29.30	0.8760
DIDN+	36.52	0.9567	30.75	0.8714	28.40	0.8054	37.37	0.9503	32.03	0.8734	29.80	0.8173	36.67	0.9639	31.82	0.9175	29.53	0.8803
RDUNet+	36.52	0.9573	30.76	0.8727	28.42	0.8077	37.34	0.9509	32.02	0.8747	29.78	0.8190	36.64	0.9640	31.75	0.9171	29.44	0.8795

TABLE 4. Number of times that an algorithm reaches the best or second-best value in the case of grayscale and color datasets.

Metric	Place	Grasyscale models						Color models					
		DIDN	RDUNet	RDN+	DHDN+	DIDN+	RDUNet+	RDN	RDUNet	RDN+	DHDN+	DIDN+	RDUNet+
PSNR	best	0	0	2	0	6	3	0	0	1	0	6	3
	second-best	2	1	1	0	3	4	1	0	2	0	2	4
SSIM	best	0	0	N/A	1	2	6	N/A	0	N/A	1	2	6
	second-best	0	3	N/A	3	0	2	N/A	5	N/A	1	0	3

TABLE 5. Weighted mean,considering the size of the studied datasets, of PSNR and SSIM measures computed from Tables 2 and 3.

Methods	Grasyscale						Color					
	$\sigma = 10$		$\sigma = 30$		$\sigma = 50$		$\sigma = 10$		$\sigma = 30$		$\sigma = 50$	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
BM3D	33.69	0.9161	28.22	0.7831	26.01	0.7008	35.05	0.9395	28.94	0.8324	25.99	0.7454
U-Net	33.78	0.9213	28.54	0.8019	26.39	0.7271	34.78	0.9461	29.47	0.8625	26.98	0.7934
DnCNN	34.22	0.9259	28.78	0.8064	26.63	0.7304	36.22	0.9562	30.42	0.8769	28.17	0.8203
IRCNN	34.09	0.9252	28.69	0.8051	26.59	0.7284	36.01	0.9547	30.38	0.8777	27.90	0.8159
FFDNet	34.10	0.9257	28.83	0.8103	26.72	0.7365	36.03	0.9554	30.56	0.8800	28.15	0.8199
RDN	34.39	N/A	29.05	N/A	26.88	N/A	36.69	N/A	31.36	N/A	28.99	N/A
DHDN	33.83	0.9211	29.03	0.8183	26.90	0.7429	36.41	0.9579	31.07	0.8889	28.71	0.8360
DIDN	34.37	0.9283	29.08	0.8164	26.96	0.7451	36.62	0.9593	31.38	0.8945	29.06	0.8440
RDUNet	34.36	0.9294	29.08	0.8185	26.96	0.7479	36.61	0.9597	31.35	0.8950	29.04	0.8449
U-Net+	33.88	0.9225	28.60	0.8038	26.45	0.7296	35.06	0.9480	29.63	0.8661	27.11	0.7990
RDN+	34.41	N/A	29.07	N/A	26.90	N/A	36.73	N/A	31.42	N/A	29.05	N/A
DHDN+	33.91	0.9227	29.07	0.8197	26.94	0.7443	36.55	0.9590	31.24	0.8923	28.90	0.8405
DIDN+	34.41	0.9285	29.12	0.8171	26.99	0.7462	36.70	0.9596	31.47	0.8957	29.16	0.8459
RDUNet+	34.39	0.9297	29.11	0.8193	26.99	0.7491	36.68	0.9600	31.43	0.8961	29.12	0.8465

We can observe a clear difference in complexity between classic models versus more recent models. Considering that the number of parameters in the RDN model is not that large, it performs too many operations.

This is mainly because it does not use sub-sampling and up-sampling operations, keeping the image with the same spatial dimension during the whole denoising process. On the other hand, the DIDN and DHDN models have a similar

TABLE 6. Number of times that an algorithm reaches the best or second-best value according to Table 5.

Metric	Place	RDN	DIDN	RDUNet	RDN+	DHDN+	DIDN+	RDUNet+
PSNR	best	0	0	0	2	0	5	1
	second-best	1	1	1	0	0	1	4
SSIM	best	N/A	0	0	N/A	1	0	5
	second-best	N/A	0	3	N/A	0	2	1

TABLE 7. Number of trainable parameters, multiply-accumulate (MAC) operations on 64 × 64 color images.

Model	DnCNN	FFDNet	U-Net	RDN	DIDN	DHDN	RDUNet
Parameters	558k	854k	31M	22M	165M	168M	166M
MACs	2G	876M	15G	90G	70G	64G	48G

complexity in the number of parameters and the number MACs. The RDUNet model has a similar amount of parameters compared to the DHDN and DIDN models. However, the number of operations it performs is lower, 25% and 31% compared to the DHDN and DIDN models, respectively. This reduction in the number of operations did not affect the algorithm's performance and kept it competitive with more complex models such as DIDN and DHDN. This is mainly because it does not use sub-sampling and up-sampling operations, keeping the image with the same spatial dimension during the whole denoising process. On the other hand, the DIDN and DHDN models have a similar complexity in the number of parameters and the number MACs. The RDUNet model has a similar amount of parameters compared to the DHDN and DIDN models. However, the number of op.

VI. LIMITATIONS OF THE STUDY

The main limitation of the proposed model is one needs to train the model for each type of noise. In particular, in the study presented in this work the model was trained for additive Gaussian noise, with the advantage that the model does not require information about the level of the noise. Other limitations are that the number of parameters and the number of operations involved during the forward mapping is high. This can be a drawback for some applications, for example: in real time video processing, in large images, or when using mobile devices in type of task. These limitations will become the future work of this study.

VII. CONCLUSION

In this work, we presented a residual dense neural network for image denoising which is based on the densely connected hierarchical network. Instead of using standard convolution-maxpool blocks, we used densely connected convolutional layers to reuse the feature maps and local residual learning to avoid the vanishing gradient problem and speed up the learning process. Furthermore, we adopted a global residual network strategy in order to take advantage of the structure that is present in the degraded image. Therefore, we modeled the residual noise, rather than modeling the clean image directly. An advantage of our proposal, in the case of additive Gaussian noise, is that it does not require information about the noise level in the degraded image. Experiments on several datasets demonstrated that our proposed network is competitive with state of the art methods and outperforms conventional methods.

REFERENCES

[1] 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)*, San Diego, CA, USA, vol. 2, Jun. 2005, pp. 60–65.

[2] 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.

[3] 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.*, Columbus, OH, USA, Jun. 2014, pp. 2862–2869.

[4] W. Zuo, K. Zhang, and L. Zhang, *Convolutional Neural Networks for Image Denoising and Restoration*. Cham, Switzerland: Springer, 2018, pp. 93–123, doi: 10.1007/978-3-319-96029-6_4.

[5] B. Park, S. Yu, and J. Jeong, "Densely connected hierarchical network for image denoising," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Long Beach, CA, USA, Jun. 2019, pp. 2095–2103.

[6] S. Yu, B. Park, and J. Jeong, "Deep iterative down-up CNN for image denoising," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Long Beach, CA, USA, Jun. 2019, pp. 2095–2103.

[7] C. Tian, L. Fei, W. Zheng, Y. Xu, W. Zuo, and C.-W. Lin, "Deep learning on image denoising: An overview," *Neural Netw.*, vol. 131, pp. 251–275, Nov. 2020.

[8] T. Wang, M. Sun, and K. Hu, "Dilated deep residual network for image denoising," in *Proc. IEEE 29th Int. Conf. Tools with Artif. Intell. (ICTAI)*, Boston, MA, USA, Nov. 2017, pp. 1272–1279.

[9] S. Bera, A. Lahiri, and P. K. Biswas, "A lightweight convolutional neural network for image denoising with fine details preservation capability," 2019, *arXiv:1903.09520*. [Online]. Available: http://arxiv.org/abs/1903.09520

[10] G. Vaksman, M. Elad, and P. Milanfar, "LIDIA: Lightweight learned image denoising with instance adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 524–525.

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

[12] P. Liu, H. Zhang, K. Zhang, L. Lin, and W. Zuo, "Multi-level wavelet-CNN for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Salt Lake, UT, USA, Jun. 2018, pp. 773–782.

[13] V. Jain and S. Seung, "Natural image denoising with convolutional networks," in *Advances in Neural Information Processing Systems 21*, vol. 21, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. Vancouver, BC, Canada: Curran Associates, 2009, pp. 769–776.

[14] 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.*, Providence, RI, USA, Jun. 2012, pp. 2392–2399.
[15] 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.
[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, Lille, France, 2015, pp. 448–456.
[17] Y. Tai, J. Yang, X. Liu, and C. Xu, "MemNet: A persistent memory network for image restoration," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 4549–4557.
[18] K. Zhang, W. Zuo, and L. Zhang, "FFDNet: Toward a fast and flexible solution for CNN-based image denoising," *IEEE Trans. Image Process.*, vol. 27, no. 9, pp. 4608–4622, Sep. 2018.
[19] J. Chen, J. Chen, H. Chao, and M. Yang, "Image blind denoising with generative adversarial network based noise modeling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 3155–3164.
[20] W. Kumwilaisak, T. Piriyatharawet, P. Lasang, and N. Thatphithakkul, "Image denoising with deep convolutional neural and multi-directional long short-term memory networks under Poisson noise environments," *IEEE Access*, vol. 8, pp. 86998–87010, 2020.
[21] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jan. 21, 2020, doi: 10.1109/TPAMI.2020.2968521.
[22] R. Timofte, R. Rothe, and L. Van Gool, "Seven ways to improve example-based single image super resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 1865–1873.
[23] S.-F. Wang, W.-K. Yu, and Y.-X. Li, "Multi-wavelet residual dense convolutional neural network for image denoising," *IEEE Access*, vol. 8, pp. 214413–214424, 2020.
[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[25] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 2261–2269.
[26] H. Ren, M. El-Khamy, and J. Lee, "DN-ResNet: Efficient deep residual network for image denoising," in *Proc. sian Conf. Comput. Vis.*, Perth, WA, Australia: Springer, 2018, pp. 215–230.
[27] Y. Zhang, L. Sun, C. Yan, X. Ji, and Q. Dai, "Adaptive residual networks for high-quality image restoration," *IEEE Trans. Image Process.*, vol. 27, no. 7, pp. 3150–3163, Jul. 2018.
[28] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, Dec. 2015, pp. 1026–1034.
[29] A. Levin and B. Nadler, "Natural image denoising: Optimality and inherent bounds," in *Proc. CVPR*, Providence, RI, USA, Jun. 2011, pp. 2833–2840.
[30] E. Agustsson and R. Timofte, "NTIRE 2017 challenge on single image super-resolution: Dataset and study," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Honolulu, HI, USA, Jul. 2017, pp. 114–125.
[31] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Represent.*, New Orleans, LA, USA, 2018. [Online]. Available: https://openreview.net/pdf?id=Bkg6RiCqY7 and https://openreview.net/forum?id=Bkg6RiCqY7
[32] S. Roth and M. J. Black, "Fields of experts: A framework for learning image priors," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, San Diego, CA, USA, vol. 2, Jun. 2005, pp. 860–867.
[33] R. Franzen. (Apr. 1999). *Kodak Lossless True Color Image Suite*. [Online]. Available: http://r0k.us/graphics/kodak
[34] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 5197–5206.
[35] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Color image denoising via sparse 3D collaborative filtering with grouping constraint in luminance-chrominance space," in *Proc. IEEE Int. Conf. Image Process.*, San Antonio, TX, USA, Sep. 2007, pp. I-313–I-316.
[36] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep CNN denoiser prior for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 2808–2817.
[37] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of PSNR in image/video quality assessment," *Electron. Lett.*, vol. 44, no. 13, pp. 800–801, Jun. 2008.
[38] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

JAVIER GURROLA-RAMOS received the B.Eng. degree in computer engineer from UAA Aguascalientes Mexico in 2016 and the M.Sc. degree in computer science and industrial mathematics from the Mathematics Research Center, Guanajuato, Mexico, in 2018, where he is currently pursuing the Ph.D. degree in computer science. His research interests lie in the areas of machine learning, optimization, image processing, and pattern recognition.

OSCAR DALMAU received the B.Sc.Ed. degree in mathematics from ISP, Manzanillo, Cuba, in 1989, and the M.Sc. degree in computer science and industrial mathematics and the Ph.D. degree in computer science from the Mathematics Research Center (CIMAT), Guanajuato, Mexico, in 2004 and 2010, respectively. He is currently with CIMAT. His research interests lie in the areas of machine learning, optimization, image processing, and computer vision.

TERESA E. ALARCÓN received the Engineering degree in automated systems of management from the Sergo Ordzhonikidze Russian State Geological Prospecting University, Russia, in 1989, the master's degree in digital image processing from the José Antonio Echeverría Polytechnic Institute, Cuba, in 1999, and the Ph.D. degree in computer science from the Center for Mathematics Research, Guanajuato, Mexico, in 2007. She is currently an Associate Professor with the Department of Computational Sciences and Engineering, University of Guadalajara, Valley Campus, Ameca, Jalisco. Her field of research is digital image processing, including filtering, segmentation, and pattern recognition topics.