sensors

MDPI

Article

Denoising Single Images by Feature Ensemble Revisited

Masud An Nur Islam Fahim, Nazmus Saqib, Shafkat Khan Siam and Ho Yub Jung *

Department of Computer Engineering, Chosun University, Gwangju 61452, Korea
* Correspondence: hoyub@chosun.ac.kr

Abstract: Image denoising is still a challenging issue in many computer vision subdomains. Recent studies have shown that significant improvements are possible in a supervised setting. However, a few challenges, such as spatial fidelity and cartoon-like smoothing, remain unresolved or decisively overlooked. Our study proposes a simple yet efficient architecture for the denoising problem that addresses the aforementioned issues. The proposed architecture revisits the concept of modular concatenation instead of long and deeper cascaded connections, to recover a cleaner approximation of the given image. We find that different modules can capture versatile representations, and a concatenated representation creates a richer subspace for low-level image restoration. The proposed architecture's number of parameters remains smaller than in most of the previous networks and still achieves significant improvements over the current state-of-the-art networks.

Keywords: feature ensemble; image denoising; SSIM

Citation: Fahim, M.A.N.I.; Saqib, N.; Siam, S.K.; Jung, H. Denoising Single Images by Feature Ensemble Revisited. *Sensors* **2022**, *22*, 7080. https://doi.org/10.3390/s22187080

Academic Editors: Kai Zhang and Dongwei Ren

Received: 27 July 2022
Accepted: 16 September 2022
Published: 19 September 2022

1. Introduction

Image denoising is a classic problem in the low-level vision domain. A given image $\mathcal{X}$ goes through the following mapping to create its noisy counterpart.

$$\mathcal{Y} = \mathcal{X} + \mathcal{N}$$

Here, $\mathcal{Y}$ is the noisy observation, where $\mathcal{N}$ is the additive noise on a clean image $\mathcal{X}$. Denoising is an ill-posed problem, with no direct means to separate the source image and corresponding noise. Hence, researchers follow the best possible approximation of $\mathcal{X}$ from $\mathcal{Y}$ with corresponding algorithmic strategies.

Typical methods without machine learning involve employing efficient filtering techniques such as NLM [1], BM3D [2], median [3], Weiner [4], etc. Due to their limited generalization capability, additional knowledge-based priors or matrix properties have been integrated into these denoising strategies. However, despite certain improvements with prior-based methods, many concerns remain unresolved, such as holistic fidelity or the choice of priors.

Convolution neural network (CNN) denoising methods later offered an unprecedented improvement over the previous strategies through their customized learning setup. Usually, CNN methods offer better performance through brute force learning [5], tricky training strategy [6], or inverting image properties [7] by various proposals. We observed gradual improvements over the years for denoising solutions. However, these methods with pure brute force mapping sometimes face fidelity issues within challenging noisy images. Furthermore, due to the lack of generalization properties, the methods provide reconstructed images that often result in cartoonized smoothing.

In contrast, the proposed approach rebuilds a previous ensemble-oriented denoising network that can successfully estimate a cleaner image with less cartoon-like smoothing. For the design of the proposed denoising network, we carefully maximized detail restoration by providing a variety of low-level ensemble features while keeping the network relatively shallow to prevent an oversized receptive field and hallucination effects. In summary, our study has the following contributions:

- We propose a shallow ensemble approach through feature concatenation to create a large array of feature combinations for low-level image recovery.
- Due to the ensemble of multiple modules, our model successfully returns fine details compared to previous data-driven studies.
- The parameter space is relatively small compared to the contemporary methods with a computationally fast inference time.
- Finally, the proposed study shows better performance with a different range of synthetic noise and real noise without the cartoonization and hallucination effect. See Figure 1.

(**a**) (**b**) (**c**) (**d**)

Figure 1. Demonstration of our contribution. An image from the BSD68 dataset with additive white gaussian noise (AWGN) noise with σ = 50. Here, the first column shows the ground truth, followed by the inference from the DEAMNet [8], and the final column shows the proposed result. From a side-by-side comparison, the proposed method can restore the image without the hallucination effects of deeper networks. (**a**) Noisy. (**b**) Ground Truth. (**c**) DEAMNet [8]. (**d**) Proposed.

2. Related Work

2.1. Filtering Based Schemes

Traditional filtering approaches aim for handcrafted filters for noise and image separation. These studies [3,4] utilized low-pass filtering methods to extract the clean images from the noisy images. The iterative filtering approach adopted a progressive reduction for image restoration [9]. Additionally, several methods used nonlocal similar patches for noise reduction based on the similarity between the counterpart patches in the same image. For example, NLM [1] and BM3D [2] assumed a redundancy within patches from a given image for noise reduction. Nonetheless, these methods usually produce flat approximations, as the given image severely degrades the noisy image quality with a heavy noise presence.

2.2. Prior Based Schemes

Another group of studies focused on selecting priors for the model, which produce clean images when optimized. These methods reformulated the denoising problem as a maximum a posteriori (MAP)-based optimization problem, where the image prior regulated the performance of the objective function. For example, the studies [10,11] assumed sparsity as the prior for their optimization process. The primary intuition was to represent each patch separately through the help of a function. Xu et al. [12] performed real-world image denoising by proposing a trilateral weighted sparse coding scheme. Other studies [12–15] focused on rank properties to minimize their objective function. Weighted nuclear norm minimization (WNNM) [12] calculated the nuclear norm through a low-rank matrix approximation for image denoising. Additionally, there are several complex model-based derivations using graph-based regularizers for noise reduction. However, their performance degrades monotonically for noisier areas, and recovering the detailed information is sometime difficult [16–18]. Additionally, these methods generally output significantly varying results depending on their prior parameters and the respective target noise levels.

2.3. Learning Based Schemes

Due to the availability of paired data and the current success of CNN modules, data-driven schemes have achieved significant improvements in separating clean images from noisy images. Recent CNN studies [19–21] utilized the residual connection for estimating the noise removal map before inference. These studies evaluated the clean image without taking any priors regarding the structure or noise. They achieved enhanced performance by using a noncomplex architecture with repeated convolutional, batch normalization, and ReLU activation function blocks. However, these methods can fail to recover some of the detailed texture structure in the presence of heavy noise area.

Trainable nonlinear reaction diffusion (TRND) [22] used a prior in its neural network and extended the nonlinear diffusion algorithm for noise reduction. However, the method suffered from computational complexity as it required a vast number of parameters. Similarly, the nonlocal color net [23] utilized the nonlocal similarity priors for the image denoising operation. Although priors mostly aid the denoising, there are some cases where the adaptation of the priors degrades the denoising performance. Very recently, DEAMNet [8] surpassed the previous state-of-the-art results by using an adaptive consistency prior.

With the success of the DnCNN [19], two similar networks called "Formatting net" and "DiffResNet" were proposed with different loss layers [5]. Later, Bae et al. [24] proposed a residual learning strategy based on the improved performance of a learning algorithm using manifold simplification, providing significantly better performance. After that, Anwar et al. [25] proposed a cascaded CNN architecture with feature attention using a self-ensemble to boost the performance.

A few recent approaches [26,27] followed the blind denoising strategy. CBDNet [27] proposed a blind denoising network consisting of two submodules, noise estimation and noise removal, by incorporating multiple losses. However, their performance was limited by manual intervention requirements and a slightly lower performance on real-world noisy images. In comparison, FFDNet [28] achieved enhanced results by proposing the nonblind Gaussian denoising network. Consequently, RIDNet [5] utilized perceptual loss with ℓ_2 apart from the DnCNN architectures for noise removal and achieved significant success by introducing a single-stage attention denoising architecture from real and synthetic noises. Liu et al. [7] introduced GradNet by revisiting the image gradient theory of neural networks. Recently, several GAN-based approaches [29–32] were introduced through generating denoised images following either a data augmentation strategy for creating diverse training samples or a strategy based on the distribution of the clean images.

The tendency for a modern denoising machine learning scheme is to use a deeper network with complex training. However, Figure 1 shows some of the hallucination and oversmoothing problems of deeper networks. A hallucination of a windowed building can be observed in the DEAMNet [8] results in Figure 1. We believe that deeper network architecture with its overfitting tendencies is the cause of the hallucination and over-smoothing. We also suspect the PSNR minimization is contributing to oversmoothing. Thus, in this paper, we propose a variety of shallow networks for low-level feature accumulation as well as a network that finds a balance between PSNR and SSIM. We show that multiple feature ensembles from a variety of shallow networks are more appropriate for denoising image problems compared to a single deeper and complex network. The shallow architecture prevents overfitting, and the necessary statistics are obtained from the feature ensemble.

3. Methodology

3.1. Baseline Supervised Architecture

Recently, supervised model-based denoising methods embedded a similar baseline formation in their proposals [5,33]. In brief, it is possible to compartmentalize the baseline architecture in Figure 2 into three distinct modules: initial feature extraction, large intermediate blocks, and feature reconstruction. Typically, the primary module consists of a single

layer that serves the purpose of the initial feature learner or initial noise estimator. For any noisy image I_n, the representation of the initial block E_p is as follows:

$$E_p = \beta(I_n) \tag{1}$$

where β is the initial convolutional layer for basic feature training. Following that, we see the main restoration part of the given network through the help of an intermediate processor. Typically, this intermediate layer is a very long cascaded connection of the unique feature extractor units. From time to time, we often observe the presence of long residual-dense or residual-attention blocks as the backbone of such setup.

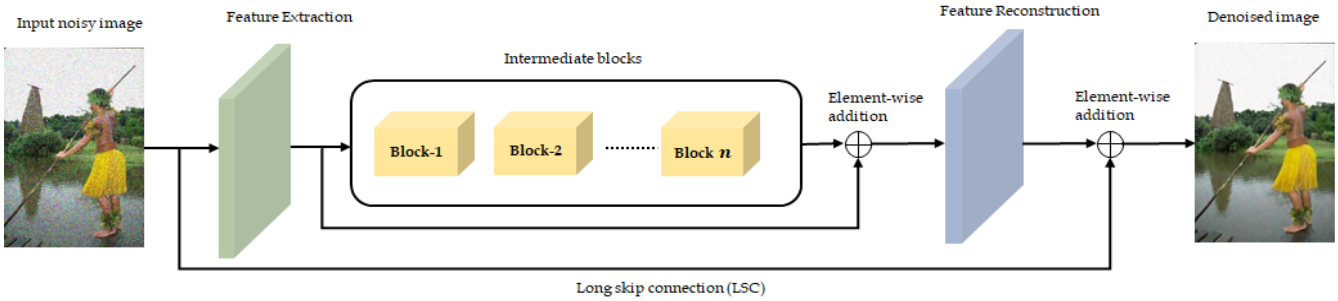


Figure 2. This figure shows the general baseline architecture for the denoising model, which usually consists of the more prolonged feature extraction phase with cascaded modules, which begin right after the initial feature collection and end with the final residual aggregation.

Now, if the intermediate block is $\mathcal{M}$, the cascaded representation of the intermediate processing stage is as follows:

$$E_i = \mathcal{M}_j(\mathcal{M}_{i-1}(.....(E_p)..)) \tag{2}$$

where $\mathcal{M}_i$ represents the ith instance of the learning stage of the intermediate block and E_i is the corresponding outcome of the intermediate layer.

The final reconstruction module operates through a residual connection followed by a consecutive final convolution. If $\mathcal{R}$ is the final reconstruction stage before the output, then the recovered image I_r is a combination of E_i, E_p, and I_n.

$$I_r = \mathcal{R}(E_i, E_p, I_n) = f_N(I_n) \tag{3}$$

Here, f_N denotes the overall neural network, I_n is the noisy input image, and I_r is the recovered image. A typical choice of the cost function for this task involves the ℓ_1 or ℓ_2 loss. There are other customized loss functions available, such as weighted-augmentation of different loss functions that integrate spatial properties or relevant regularization [5]. In general, the network is optimized by minimizing the difference from clean images.

$$\zeta(\theta) = \frac{1}{N}\sum_{i=1}^{N} ||f_N(\theta, I_n^i) - I_c^i||_1 \tag{4}$$

Here, θ is the learnable parameter, I_n^i is the noisy image, and I_c^i is the corresponding clean image. Most of the baseline network parameters are placed in the intermediate learning block.

3.2. Proposed Architecture

In contrast, we designed our network to allocate more resources to the concatenated learned features. Instead of developing a basic learning block for long cascading connections, we chose variety by proposing various individual feature learning blocks. The proposed network is focused on delivering richer and diverse low-level features. To fur-

ther reduce complexity, we avoided using an attention operation, which is typically more expensive. More details are provided in Figure 3 and the following subsection.

Initial Feature Block

Three consecutive convolution layers are used to extract the initial features for the network. The layers are equal in depth, but their kernel sizes were in descending order. The input image goes through the 5×5 convolution operation at the first layer, followed by a 3×3 convolution, and ends at the 1×1 pixelwise operation. A larger kernel size makes use of a larger neighborhood of input features and estimates the representations on larger receptive fields. By limiting the kernel size and the number of layers, the network learns to focus on the smaller receptive fields and disregards the broader view, which we argue to be less meaningful in low-level vision tasks such as denoising. Therefore, the purpose of the primary layer is to project the representation for the denoising features from a smaller receptive field into individual responses which can be further diversified in the next four block modules in Figure 3.

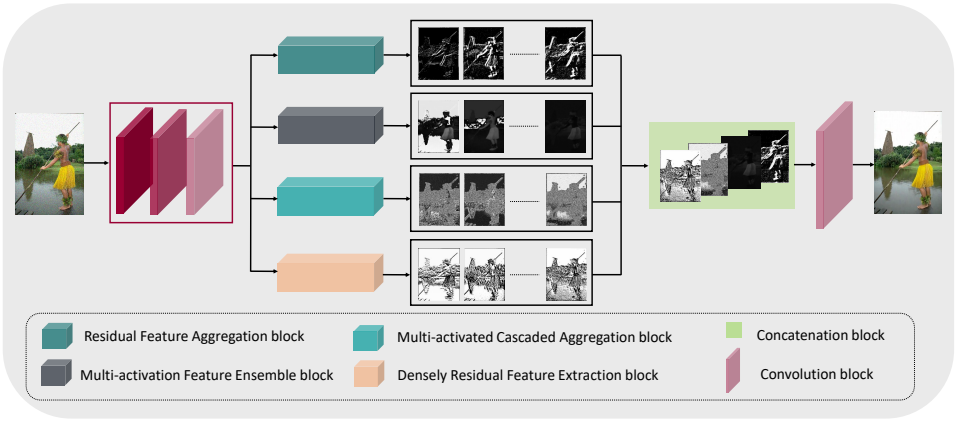


Figure 3. In the above figure, we present the overall diagram of the proposed architecture for image denoising. Our pipeline first extracts the initial feature using consecutive convolution operation, followed by the four modules for feature refinements. These modules are standing upon the customized convolution and residual setup with supportive activation functions. After refinement, we concatenate all the refined feature maps into a single layer, followed by a final dilated convolution to make the inference.

3.3. Four Modules for Feature Refinement

Before presenting the four modules for feature refinement, we cover the convolution, activation functions, and residual connections used in the modules. Even though the attention mechanism is a common choice to learn richer representations, we can still find a similar or better result without it in this study. Additionally, our selection of residual blocks was for blocks to be no longer than six consecutive connections. The fundamental operations for our modules are introduced below.

Convolution. In the internal convolution operation, our choice of kernels varied from 1×1 to 7×7. Due to such a range, our network was naturally focused on both smaller and larger receptive fields.

Activation functions. Recent advancements in nonlinear activation functions have shown that better performance is achievable through the interconnected operation of different activation representations that are compacted into a single function. Hence, we chose the SWISH [34] and MISH [35] activation representations in addition to the ReLU operations. As a result, our network learned from diverse representations obtained from various parallel activated functions.

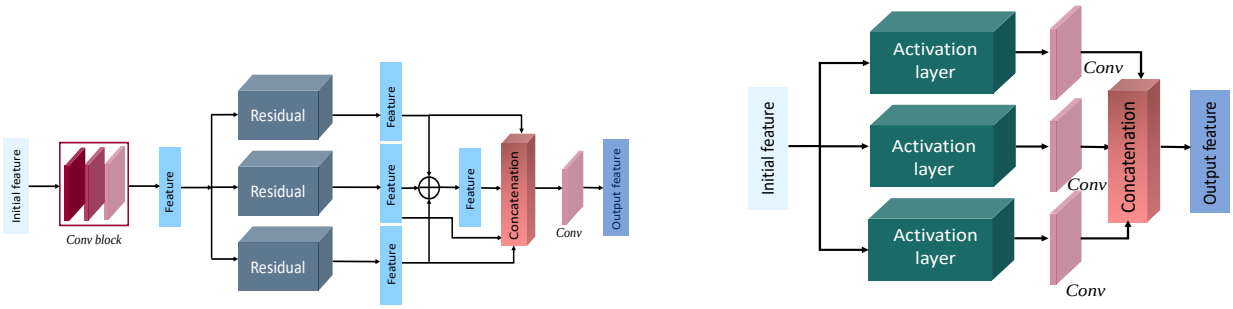
Residual connections. It is redundant to mention the efficacy of the residual connections in the vision tasks. In the literature, we can see that the customization of residual connections varies within the task. In the original ResNet paper [36], the authors included

batch-normalization between the convolution layer, followed by the ReLU layer. In our study, we used the convolution layers, which were separated by the ReLU layer. This choice of the ReLU sandwich residual connection is prevalent in regression tasks [37].

We focus on the major processing modules below with the description of the utilized blocks. We propose four processing modules that perform the refinement operations on the initial features. The following subsections cover their descriptions and the basic reasoning behind the proposed architecture.

3.3.1. Residual Feature Aggregation Module

In our residual feature aggregation module, we used the aforementioned residual blocks as our underlying design mechanism. In the construction of this module, we took inspiration from the traditional pyramid feature extraction [38] and aggregation, which has been very influential in computer vision. A typical pyramid setup is motivated by the needs for multiscale feature aggregation, which, in essence, utilizes low-frequency information along with high-frequency features. However, the subsequent downsampling process is a lossy operation by nature. To mitigate information loss for low-frequency features, we chose to employ the concurrent residual blocks on the same initial features through three different kernel sizes. Naturally, our kernel choice ranged from 1×1 to 5×5, as seen in Figure 4a. Hence, a larger kernel allowed us to learn the features from a larger area of image, while the 1×1 kernel operation allowed us to maintain the initial receptive field and make use of more high-frequency information. We aggregated the response from all three residual block to learn the overall multiscale impact of the initial features. Finally, a typical 3×3 convolution with standard depth gave us the n number of diverse representations from this module. As a result, our model can learn the important multiscale features without going through a pooling operation.



(**a**) (**b**)

Figure 4. The first two modules in our proposed architecture. The first one is the residual feature aggregation module, and the second one is the multiactivation feature ensemble module. (**a**) Residual feature aggregation module. (**b**) Multiactivation feature ensemble module.

3.3.2. Multiactivation Feature Ensemble

Activation functions are unavoidable components for neural network construction that aid the learning operation by projecting the impactful information to the next layer. Hence, widely different nonlinear functions are available as activation functions in all sorts of neural networks for various purposes. The ReLU is the most widely used activation function, which at heart is a "positive pass" filter. However, in some cases, zero-out negatives and a discontinuity in the gradient are argued to be unhelpful in the optimization process. To address some of its weaknesses, SWISH [34] and MISH [35] were proposed with smooth gradients while maintaining a similar positive-pass shape of the ReLU. A recent experiment [35] showed that these activation functions provided a smoother loss landscape than the ReLU.

Nonetheless, we incorporated all three activation functions, as seen in Figure 4b. SWISH, MISH, and ReLU activation functions were applied to the initial features, followed by a convolution layer. The subsequent responses were concatenated into a single tensor

to learn from the integrated representation of varying activation functions. No further kernels and residual blocks were utilized for this module. The initial feature results of these modules were ensembled with the responses of the other three modules, but the multiactivation functions were also integrated into the multiactivated cascaded aggregation module described in Section 3.3.3.

3.3.3. Multiactivated Cascaded Aggregation

In this module, both shallow and relatively deeper layer features were concatenated. Typically, a deep consecutive convolution operation is formulated after the initial feature extraction, and the conventional thinking is to build a deeper network for complex problems. However, we added a single convolution layer feature to complement the deeper layer features because we believed that a shallower interpretation might be more appropriate for low-level vision problems. See Figure 5.

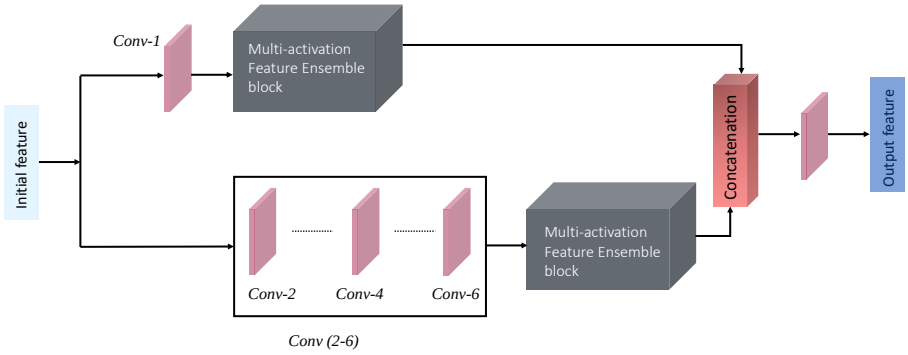


Figure 5. Multiactivated cascaded aggregation module.

For a single convolution path, a 3×3 kernel size was chosen with the same depth as the initial features. For the deeper path, five consecutive convolution layers with different kernel sizes were used. The activation functions between the layers were ReLUs, however, for both paths, a multiactivation feature ensemble was implemented as described earlier. Both the shallow and deeper responses were concatenated followed by another convolution layer.

3.3.4. Densely Residual Feature Extraction

The densely residual operation has shown great promise in both regression and classification tasks [39]. Dense residual connections are an efficient way to emphasize hierarchical representation. For this reason, we designed a densely residual module to aggregate features for the network. The proposed design in Figure 6 also utilized the concatenation between the final and previous aggregation in support of a total hierarchy concentration. A final convolution was added to combine the three concatenated features from the densely residual layers.

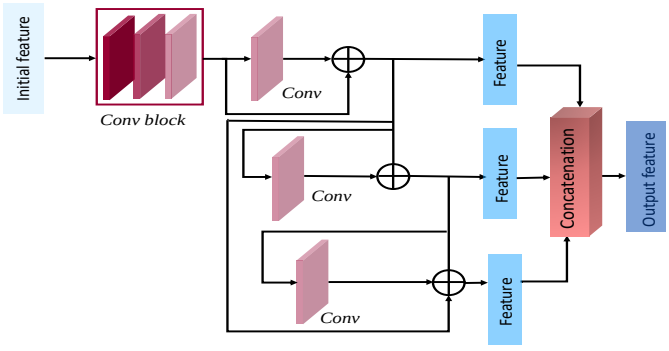


Figure 6. Densely residual feature extraction module.

After collecting and concatenating the individual responses from each of the four modules, the responses were merged by the final convolution layer with a dilation rate of 2, see the overall process in Figure 3. This layer's output contained the most refined representation for the restored image. The restored image was fed into a simple loss function consisting of ℓ_1 and ℓ_{SSIM}.

3.4. Loss Function

We used two typical loss functions, ℓ_1 and ℓ_{SSIM}, to update the parameter space. The total loss function was a simple addition of the two.

$$\ell_{total} = \ell_1 + \ell_{SSIM}. \tag{5}$$

ℓ_1 measures the distance between the ground truth, clean image and the restored image as shown in next equation.

$$\ell_1 = \frac{1}{n}\sum_{j=1}^{n} |\gamma_g - \gamma_p|. \tag{6}$$

Here, γ_g is the ground truth clean image and γ_p is the restored prediction image. The secondary component is the loss function from SSIM, which is another widely used similarity measure for images.

$$\ell_{SSIM} = \frac{1}{n}\sum_{j=1}^{n} 1 - SSIM(\gamma_g, \gamma_p) \tag{7}$$

4. Experimental Results

This section describes the overall performance of our method on both real and synthetic noisy images.

4.1. Network Implementation and Training Set

For the proposed study, we utilized a TensorFlow framework with NVIDIA GPU support. Most of the convolutional layers in our network were 3×3 kernels, apart from the specific cases where 1×1, 5×5, and 7×7 kernels in addition to the 3×3 kernels were used. For the training phase, we used the method from He et al. [40] for the initialization and the Adam optimizer with a learning rate of 10^{-4}, a typical default in many vision studies.

For the training, the DIV2K dataset was used. To enable diversity in the data flow, the typical rotation, blurring, contrast stretching, and inverse augmentation techniques were implemented. The training images were cropped into smaller patches. The noisy input images were created by perturbing the clean patches by additive white gaussian noise (AWGN) with 15, 25, and 50 standard deviations.

4.2. Testing Set

We use the BSD68, Kodak24, and Urban100 datasets for the inference comparison, where clean observations were available and noisy versions were created through the same artificial noise augmentation. The results are summarized in Table 1.

The DND, SIDD, and RN15 datasets were used to evaluate the proposed approach on images with natural noise. A brief description of the real-world noisy image dataset and the evaluation procedures are described below.

- **DND**: DND [41] is a real-world image dataset consisting of 50 real-world noisy images. However, near noise-free counterparts are unavailable to the public. The corresponding server provides the PSNR/SSIM results for the uploaded denoised images.

- **SIDD**: SIDD [42] is another real-world noisy image dataset that provides 320 pairs of noisy images and near noise-free counterparts for training. This dataset follows a similar evaluation process as for the DND dataset.
- **RN15**: RN15 [26] dataset provides 15 real-world noisy images. Due to the unavailability of the ground truths, we only present the visual result of this dataset.

Table 1. Quantitative comparison results of the competing methods with AWGN noise levels σ =15, 25, 50 on kodak24, BSD68, and Urban100. Top results are in bold, and second-best results are underlined.

Method	Metrics	$\sigma = 15$			$\sigma = 25$			$\sigma = 50$		
		BSD68	Kodak24	Urban100	BSD68	Kodak24	Urban100	BSD68	Kodak24	Urban100
BM3D [2]	PSNR	32.37	31.07	32.35	29.97	28.57	29.70	26.72	25.62	25.95
	SSIM	0.8952	0.8717	0.9220	0.8504	0.8013	0.8777	0.7676	0.6864	0.7791
WNNM [12]	PSNR	32.70	31.37	32.97	30.28	28.83	30.39	27.05	25.87	26.83
	SSIM	0.8982	0.8766	0.9271	0.8577	0.8087	0.8885	0.7775	0.6982	0.8047
DnCNN [19]	PSNR	32.86	31.73	31.86	30.06	28.92	29.25	27.18	26.23	26.28
	SSIM	0.9031	0.8907	0.9255	0.8622	0.8278	0.8797	0.7829	0.7189	0.7874
FFDNet [28]	PSNR	32.75	31.63	32.43	30.43	29.19	29.92	27.32	26.29	26.28
	SSIM	0.9027	0.8902	0.9273	0.8634	0.8289	0.8886	0.7903	0.7245	0.8057
IrCNN [20] [1]	PSNR	31.67	33.60	31.85	29.96	30.98	28.92	26.59	27.66	25.21
	SSIM	0.9318	0.9247	0.9493	0.8859	0.8799	0.9101	0.7899	0.7914	0.8168
ADNet [43]	PSNR	32.98	31.74	32.87	30.58	29.25	30.24	27.37	26.29	26.64
	SSIM	0.9050	0.8916	0.9308	0.8654	0.8294	0.8923	0.7908	0.7216	0.8073
RIDNet [5]	PSNR	32.91	31.81	33.11	30.60	29.34	30.49	27.43	26.40	26.73
	SSIM	0.9059	0.8934	0.9339	0.8672	0.8331	0.8975	0.7932	0.7267	0.8132
VDN [44]	PSNR	**33.90**	**34.81**	33.41	**31.35**	**32.38**	30.83	28.19	**29.19**	**28.43**
	SSIM	0.9243	0.9251	0.9339	0.8713	0.8842	0.8361	0.8014	0.7213	0.8212
DEAMNet [8]	PSNR	33.19	31.91	33.37	30.81	29.44	30.85	27.74	26.54	27.53
	SSIM	0.9097	0.8957	0.9372	0.8717	0.8373	0.9048	0.8057	0.7368	0.8373
Proposed	PSNR	33.85	32.90	**33.97**	31.32	30.67	**31.52**	**29.02**	28.12	28.25
	SSIM	**0.9603**	**0.9517**	**0.9621**	**0.9150**	**0.9246**	**0.9241**	**0.8831**	**0.8782**	**0.8755**

[1] The PSNR results for Kodak24 and BSD68 were obtained from the IrCNN implementation from (https://github.com/cszn/IRCNN, accessed on 10 October 2021).We want to note that our results are significantly different from the results reported in [45].

4.3. Denoising on Synthetic Noisy Images

For evaluation purposes, we considered previous state-of-the-art studies within various contexts. The evaluation procedure included two filtering methods, BM3D [2], WNNM [12], and several convolutional networks including DnCNN [19], FFDNet [28], IrCNN [20], ADNet [43], RIDNet [5], VDN [44], and DEAMNet [8].

Table 1 shows the average PSNR/SSIM scores for the quantitative comparison. From the average PSNR and SSIM score, the proposed study surpasses the previous studies with a considerable margin. We adopted three widely used datasets BSD68, Kodak24, and Urban100 with three different AWGN noise levels, 15, 50, and 50. The code for all methods used in this evaluation, including our own source code, is found in Appendix A.

For a visual comparison, Figures 7–9 from BSD68, Kodak24, and Urban100 are presented, respectively, with a noise level of 50. Figure 7 shows the "fireman" picture from the BSD68 dataset. The differences in the restoration are shown in detail with a more controlled smoothing. From Figure 8, we see that the proposed approach avoids image cartoonization and preserves details while restoring clean details. The proposed study manages to restore the structural continuity compared to other methods while preserving the appropriate color and contrast of the image. The last visual comparison for the synthetic noisy image is the "Interior" picture from the Urban100 dataset, shown in Figure 9. For a better illustration of the differences, a zoomed image of the interior wall of the place is shown, where the proposed method manages to preserve the brick's separating lines more clearly. We also

provide Figure 10, where multiple images were combined with different intensities of noise with their proposed output.

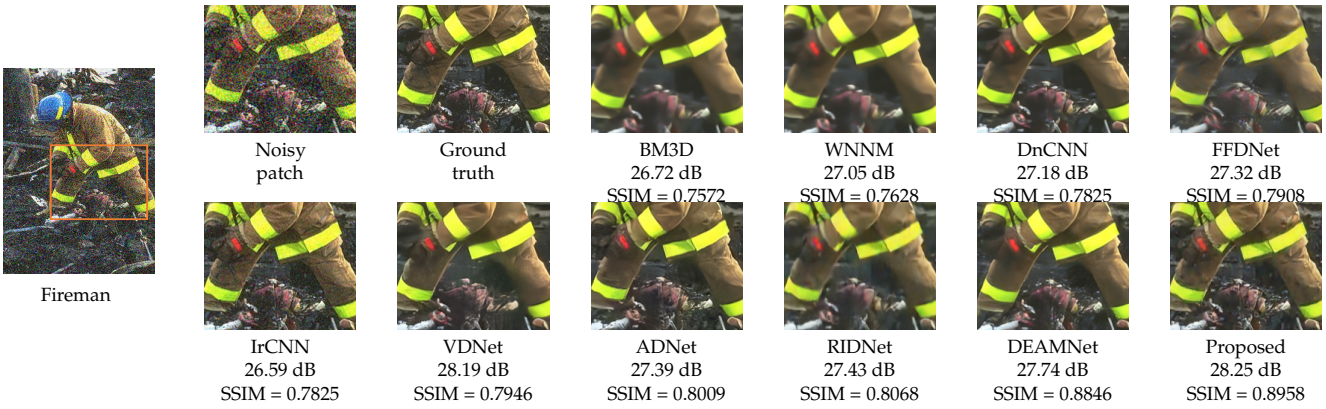


Figure 7. Visual quality comparison with PSNR and SSIM scores for "Fireman" from the BSD68 dataset with AWGN noise level $\sigma = 50$ (for best view, zooming in is recommended).

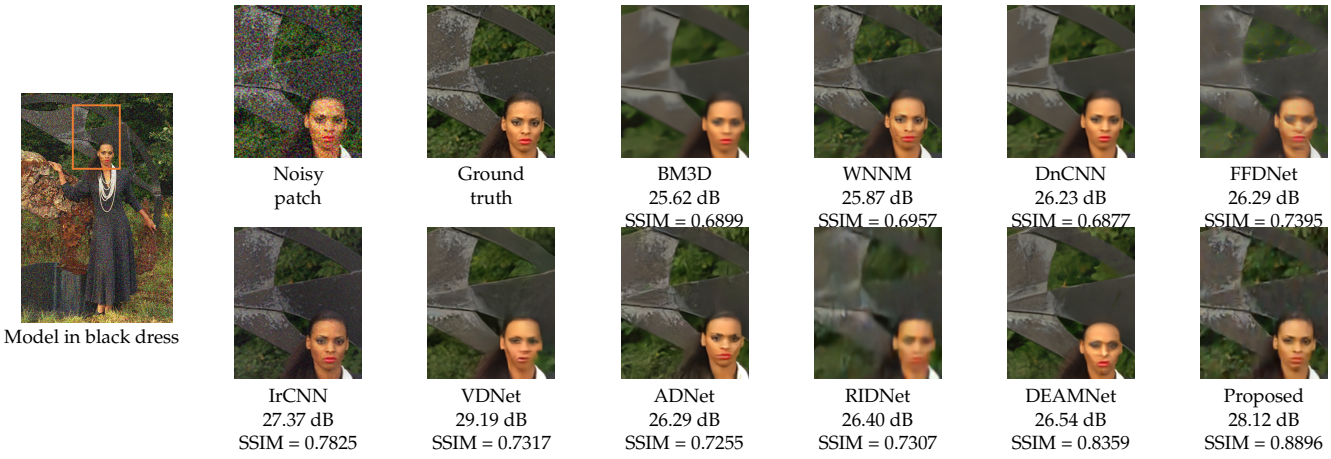


Figure 8. Visual quality comparison with PSNR and SSIM scores for "Model in black dress" from the Kodak24 dataset with AWGN noise level $\sigma = 50$ (for best view, zooming in is recommended).

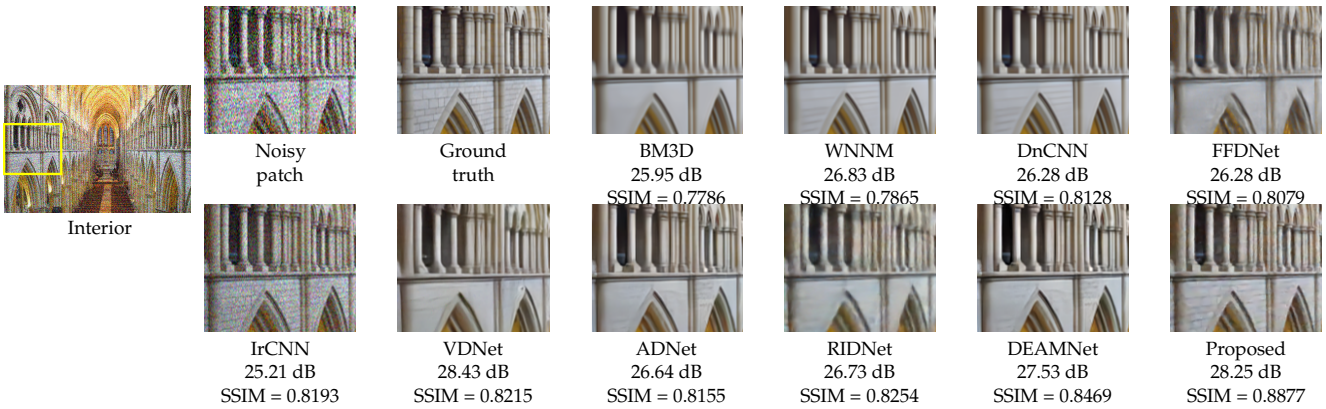


Figure 9. Visual quality comparison with PSNR and SSIM scores for "Interior" from the Urban100 dataset with AWGN noise level $\sigma = 50$ (for best view, zooming in is recommended).

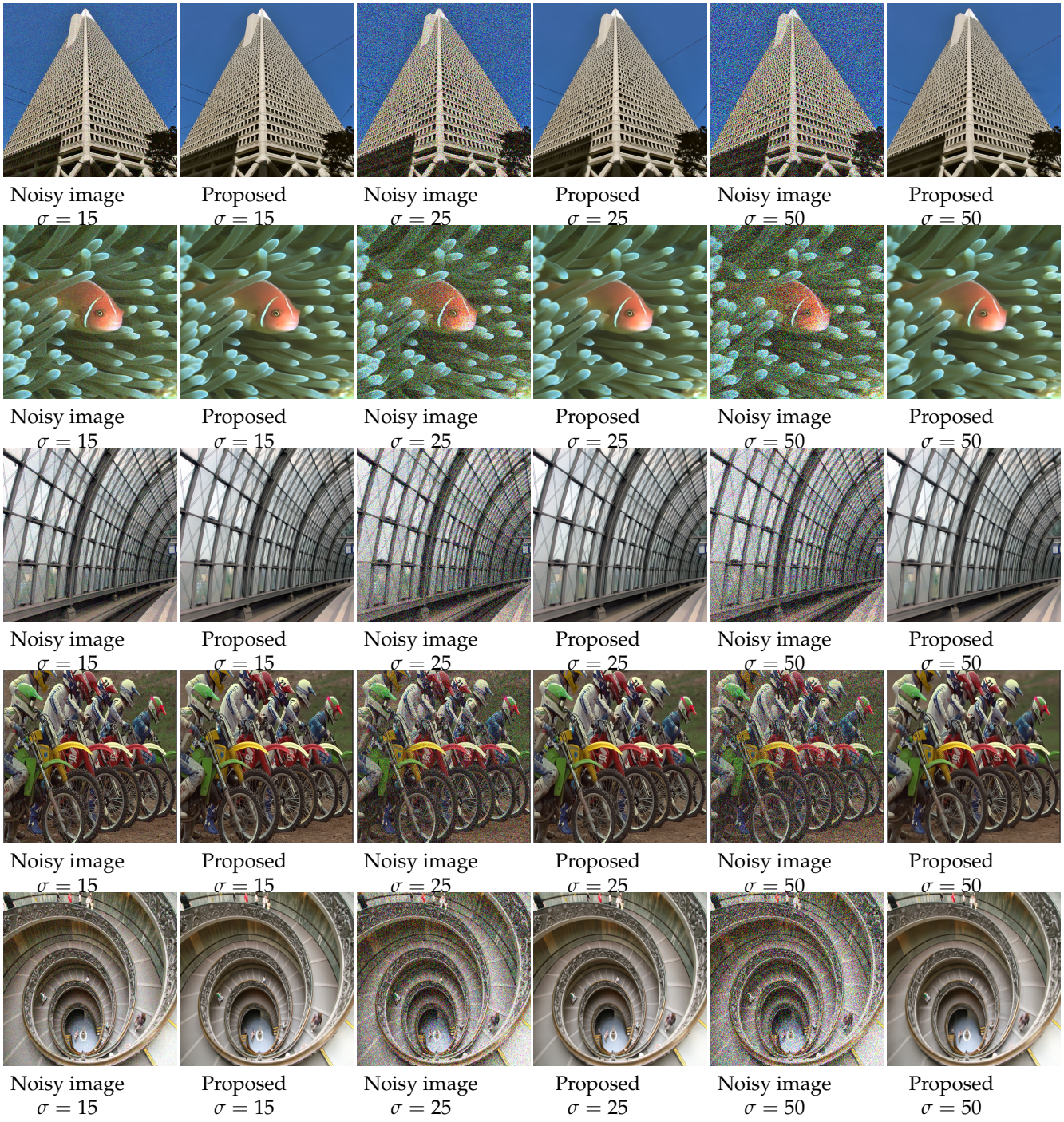


Figure 10. Sample results for different datasets for $\sigma = 15$, 25, and 50 (for best view, zooming in is recommended).

4.4. Denoising on Real-World Noisy Images

The results for real-world noisy image restoration are presented in Table 2. Natural noise removal is challenging because the convoluted noises are not signal independent and vary within the spatial neighborhood.

Table 2. Real-image denoising results of several existing methods on SIDD and DnD dataset. Top results are in bold, and second best results are underlined.

Dataset	Metrics	BM3D	DnCNN	FFDNet	VDN	RIDNet	DEAMNet	Proposed
SIDD [42]	PSNR	25.65	23.66	29.30	39.26	37.87	<u>39.35</u>	**39.55**
	SSIM	0.685	0.583	0.694	0.944	0.943	<u>0.955</u>	**0.964**
DnD [41]	PSNR	34.51	32.43	37.61	39.38	39.25	<u>39.63</u>	**39.76**
	SSIM	0.8507	0.7900	0.9115	0.9518	0.9528	<u>0.9531</u>	**0.9617**

We chose three real noisy image datasets, the SIDD benchmark [42], the DnD benchmark [41], and RN15 [26], to analyze the generalization capability of our proposed method. For the SIDD and DnD benchmarks, the clean counterpart images are not openly distributed. Hence, the presented PSNR/SSIM in Table 2 was obtained by uploading the results into the corresponding server. For the RN15 dataset, there is no benchmark utility. Table 2 represents the comparative performance for both SIDD and DnD benchmarks. Among the existing methods, VDN [44] and DEAMNet [8] perform well. However, our method achieves a better result among the existing methods for both the real and synthetic noises.

To demonstrate the performance of our method with real images, we also provide some visual comparisons in Figures 11–13 on the SIDD, DND, and RN15 datasets, respectively. For a visual comparison on real noisy images, we included the recent VDN [44], RIDNet [5], and DEAMNet [8]. The visual comparison shows that our method tends to avoid cartoonization while effectively removing noise, suppressing artifacts, and preserving object edges. Overall, the qualitative and quantitative comparisons display an effective performance on all fronts.

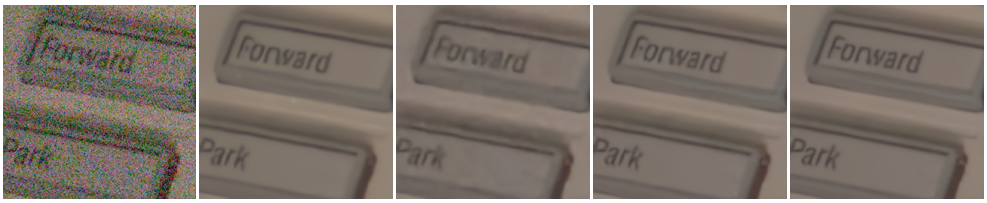


Figure 11. Visual quality comparison with PSNR and SSIM scores for the SIDD dataset with real noises (for best view, zooming in is recommended).

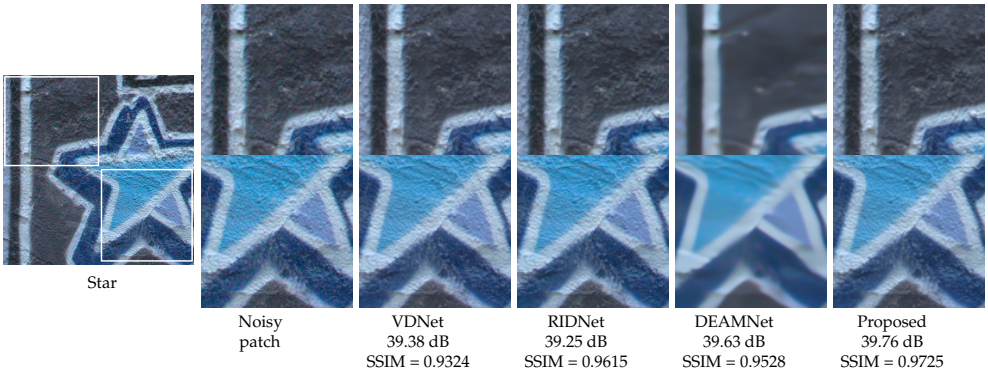


Figure 12. Visual quality comparison with PSNR and SSIM scores for "Star" from the DnD dataset with real noises (for best view, zooming in is recommended).

Figure 13. Visual quality comparison for "Dog" and "Glass" from the RN15 dataset. RN15 dataset is a set of real noise images without the clean image counterparts (for best view, zooming in is recommended).

4.5. Computational Complexity

This section provides a comparison of the computational complexity through Table 3. The table represents the average running times for the three different image sizes 256×256, 512×512, and 1024×1024. In addition, we present the parameter counts of the compared methods. Apart from BM3D [2], we report the model-specific computation time. In this comparison, we considered BM3D [2], DnCNN [19], WNNM [12], IrCNN [20], FFDNet [28], AINDNet [33], ADNet [43], VDN [44], RIDNet [5], and DEAMNet [8]. In Table 3, our method's computation time is only slightly longer than the earlier DnCNN, IrCNN, FFDNet, ADNet, and VDN. In terms of parameter counting, the proposed study is significantly smaller than the recent RIDNet [5], AINDNet [33], VDN [44], and DEAMNet [8].

Table 3. Running time (in seconds) and parameter comparison.

Method	Size 256^2	Size 512^2	Size 1024^2	Parameters
BM3D [2]	0.76	3.12	12.82	-
WNNM [12]	210.26	858.04	3603.68	-
DnCNN [19]	0.01	0.05	0.16	558 k
IrCNN [20]	0.012	0.038	0.146	-
FFDNet [28]	0.01	0.05	0.11	490 k
AINDNet [33]	0.05	0.03	0.80	13,764 k
ADNet [43]	0.02	0.06	0.20	519 k
VDN [44]	0.04	0.07	0.19	7817 k
RIDNet [5]	0.07	0.21	0.84	1499 k
DEAMNet [8]	0.05	0.19	0.73	2225 k
Proposed	0.031	0.11	0.42	846 k

4.6. Ablation Study on Modules

In this section, we provide an ablation study based on the effect on our modules' correlation. We used four different modules, which work separately and generate various features. These different features cannot be considered separately as clean images. However, if we concatenate them together as the proposed method described, we can obtain a clean image. In Table 4, the modules are the residual feature aggregation block (RFA), multiactivation feature ensemble block (MFE), multiactivated cascaded aggregation block (MCA), and densely residual feature extraction block (DRFE). We removed each module separately and calculated the PSNR and SSIM for three different datasets. Here, we can observe that the PSNR value drops every time a module is removed. For the multiactivation feature ensemble block (MFE), the value of the PSNR drops the most, and for the module multiactivated cascaded aggregation block (MCA), the SSIM value drops the most. In Figure 14, we represent the output of these four modules separately with the ground truth and our proposed method's output.

Table 4. Removing different modules from the ensemble and comparing their results on different datasets.

Dataset	PSNR and SSIM	RFA Module Removed	MFE Module Removed	MCA Module Removed	DRFE Module Removed
BSD68	33.85 0.9603	30.65 0.885	28.66 0.783	29.30 0.824	30.26 0.855
Kodak24	32.90 0.9517	29.51 0.7507	27.43 0.6900	28.61 0.8115	30.38 0.8518
Urban100	33.97 0.9621	30.48 0.7824	27.75 0.6192	30.54 0.7822	31.38 0.8766

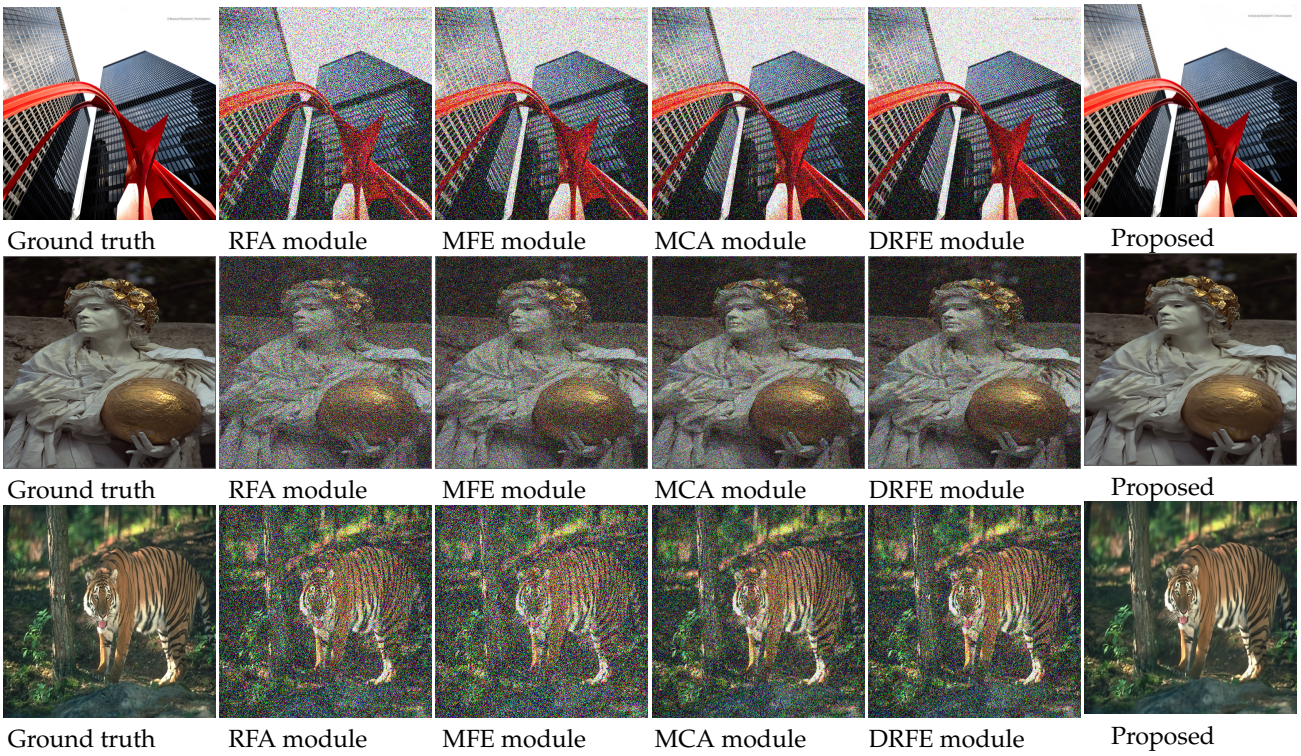


Figure 14. Sample results for all four modules separately (for best view, zooming in is recommended).

5. Conclusions

In this paper, the basic strategy for the low-level denoising problem was to gather a variety of low-level features while keeping the interpretation simple by implementing relatively shallow layers. We argued that for low-level vision tasks, the principle of Occam's razor was more appropriate, and accordingly, we designed a network that focused on gathering a variety of low-level evidence rather than providing a deep explanation of the evidence. Thus, we revisited the feature ensemble approach for the image denoising problem. Our study offered a new model which concatenated different modules for creating large and varying feature maps. To enhance the performance of our network, we utilized different kernel sizes, residual and densely residual connections, and avoided deep unimodule cascaded aggregation. We carefully designed four different modules for our study, where each helped to restore different spatial properties. Finally, we validated our network with natural and synthetic noisy images. Extensive comparisons showed the overall efficiency of the proposed study. We observed that although our SSIM scores were much higher across the board, the PSNR scores were not the best in the comparison. Our model extracted a variety of shallow features from the image; however, for higher PSNR evaluation, a deeper network may be desirable. In future work, we are planning to apply a self-supervised strategy in training procedures using the same ensemble of shallow

networks. The different versions of the noisy input images are planned to be used during the denoising self-supervised training.

Author Contributions: M.A.N.I.F. designed the main experiment. N.S. and M.A.N.I.F. wrote the main parts of the paper. S.K.S. conducted the related experiments with different datasets. H.Y.J. did the overall revision and presentation. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2021R1A2C1009776).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets that support the findings of this study are openly available in BSD at https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/, Kodak at http://r0k.us/graphics/kodak/, Urban at https://github.com/majedelhelou/denoising_datasets/tree/main/CUrban100/, DND at https://noise.visinf.tu-darmstadt.de/, SIDD at https://www.eecs.yorku.ca/~kamel/sidd/benchmark.php, and RN15 at https://demo.ipol.im/demo/125/archive/.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

In the appendix, we list the open source code we used for the evaluations. First, the code for the proposed network can be found at (https://github.com/cvlabchosun/denoising_ensemble, accessed on 7 September 2022). The github repository also contains the SSIM and PSNR calculations for Table 1, as well as the denoised images with a noise level of 50 for different datasets. We used the built-in BM3D [2] python library to generate the BM3D images. WNNM [12] was evaluated using (https://github.com/csjunxu/WNNM_CVPR2014, accessed on 27 November 2017). DnCNN [19] was evaluated using (https://github.com/cszn/DnCNN, accessed on 10 October 2021). The FFDNet [28] results were from (https://github.com/cszn/FFDNet, accessed on 10 October 2021). The IrCNN [20] results were from (https://github.com/cszn/IRCNN, accessed on 10 October 2021). (https://github.com/cqray1990/ADNet, accessed on 17 January 2020). was used for the ADNet [43] results. The VDN [44] results were from (https://github.com/zsyOAOA/VDNet, accessed on 29 June 2021). Finally, the DEAMNet [8] results were obtained from (https://github.com/chaoren88/DeamNet, accessed on 23 June 2021).

References

1. Buades, A.; Coll, B.; Morel, J.M. A non-local algorithm for image denoising. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 2, pp. 60–65. [CrossRef]
2. Dabov, K.; Foi, A.; Katkovnik, V.; Egiazarian, K. Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering. *IEEE Trans. Image Process.* **2007**, *16*, 2080–2095. [CrossRef] [PubMed]
3. Chen, T.; Ma, K.K.; Chen, L.H. Tri-state median filter for image denoising. *IEEE Trans. Image Process.* **1999**, *8*, 1834–1838. [CrossRef] [PubMed]
4. Chen, J.; Benesty, J.; Huang, Y.; Doclo, S. New insights into the noise reduction Wiener filter. *IEEE Trans. Audio Speech Lang. Process.* **2006**, *14*, 1218–1234. [CrossRef]
5. Anwar, S.; Barnes, N. Real Image Denoising with Feature Attention. In Proceedings of the IEEE International Conference on Computer Vision (ICCV-Oral), Seoul, Korea, 27 October–2 November 2019.
6. Ouyang, J.; Adeli, E.; Pohl, K.M.; Zhao, Q.; Zaharchuk, G. Representation Disentanglement for Multi-modal MR Analysis. *arXiv* **2021**, arXiv:2102.11456.
7. Li, P.; Chen, B.; Ouyang, W.; Wang, D.; Yang, X.; Lu, H. GradNet: Gradient-guided network for visual object tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 6162–6171.
8. Ren, C.; He, X.; Wang, C.; Zhao, Z. Adaptive Consistency Prior Based Deep Network for Image Denoising. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021; pp. 8596–8606.

9. Knaus, C.; Zwicker, M. Progressive Image Denoising. *IEEE Trans. Image Process.* **2014**, *23*, 3114–3125. [CrossRef]
10. Xie, Q.; Zhao, Q.; Meng, D.; Xu, Z.; Gu, S.; Zuo, W.; Zhang, L. Multispectral Images Denoising by Intrinsic Tensor Sparsity Regularization. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1692–1700. [CrossRef]
11. Elad, M.; Aharon, M. Image Denoising Via Sparse and Redundant Representations Over Learned Dictionaries. *IEEE Trans. Image Process.* **2006**, *15*, 3736–3745. [CrossRef]
12. Gu, S.; Zhang, L.; Zuo, W.; Feng, X. Weighted Nuclear Norm Minimization with Application to Image Denoising. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 2862–2869. [CrossRef]
13. Xie, Y.; Gu, S.; Liu, Y.; Zuo, W.; Zhang, W.; Zhang, L. Weighted Schatten p-Norm Minimization for Image Denoising and Background Subtraction. *IEEE Trans. Image Process.* **2016**, *25*, 4842–4857. [CrossRef]
14. Xie, T.; Li, S.; Sun, B. Hyperspectral Images Denoising via Nonconvex Regularized Low-Rank and Sparse Matrix Decomposition. *IEEE Trans. Image Process.* **2019**,*29*, 44–56. [CrossRef]
15. Xu, J.; Zhang, L.; Zhang, D.; Feng, X. Multi-channel weighted nuclear norm minimization for real color image denoising. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 1096–1104.
16. Pang, J.; Cheung, G. Graph Laplacian Regularization for Image Denoising: Analysis in the Continuous Domain. *IEEE Trans. Image Process.* **2017**, *26*, 1770–1785. [CrossRef]
17. Xu, L.; Lu, C.; Xu, Y.; Jia, J. Image Smoothing via L0 Gradient Minimization. *ACM Trans. Graph. SIGGRAPH Asia* **2011**, *30*, 1–12.
18. Mahdaoui, A.E.; Ouahabi, A.; Moulay, M.S. Image Denoising Using a Compressive Sensing Approach Based on Regularization Constraints. *Sensors* **2022**, *22*, 2199. [CrossRef] [PubMed]
19. Zhang, K.; Zuo, W.; Chen, Y.; Meng, D.; Zhang, L. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Trans. Image Process.* **2017**, *26*, 3142–3155. [CrossRef] [PubMed]
20. Zhang, K.; Zuo, W.; Gu, S.; Zhang, L. Learning Deep CNN Denoiser Prior for Image Restoration. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3929–3938.
21. Song, Y.; Zhu, Y.; Du, X. Dynamic Residual Dense Network for Image Denoising. *Sensors* **2019**, *19*, 3809. [CrossRef] [PubMed]
22. Chen, Y.; Pock, T. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1256–1272. [CrossRef] [PubMed]
23. Lefkimmiatis, S. Non-local color image denoising with convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3587–3596.
24. Bae, W.; Yoo, J.; Chul Ye, J. Beyond deep residual learning for image restoration: Persistent homology-guided manifold simplification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Honolulu, HI, USA, 21–26 July 2017; pp. 145–153.
25. Anwar, S.; Huynh, C.P.; Porikli, F. Chaining identity mapping modules for image denoising. *arXiv* **2017**, arXiv:1712.02933.
26. Lebrun, M.; Colom, M.; Morel, J.M. The noise clinic: A universal blind denoising algorithm. In Proceedings of the 2014 IEEE International Conference on Image Processing (ICIP), Paris, France, 27–30 October 2014; pp. 2674–2678. [CrossRef]
27. Guo, S.; Yan, Z.; Zhang, K.; Zuo, W.; Zhang, L. Toward convolutional blind denoising of real photographs. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 1712–1722.
28. Zhang, K.; Zuo, W.; Zhang, L. FFDNet: Toward a fast and flexible solution for CNN-based image denoising. *IEEE Trans. Image Process.* **2018**, *27*, 4608–4622. [CrossRef]
29. Chen, S.; Shi, D.; Sadiq, M.; Cheng, X. Image Denoising With Generative Adversarial Networks and its Application to Cell Image Enhancement. *IEEE Access* **2020**, *8*, 82819–82831. [CrossRef]
30. Park, H.S.; Baek, J.; You, S.K.; Choi, J.K.; Seo, J.K. Unpaired image denoising using a generative adversarial network in X-ray CT. *IEEE Access* **2019**, *7*, 110414–110425. [CrossRef]
31. Li, W.; Wang, J. Residual Learning of Cycle-GAN for Seismic Data Denoising. *IEEE Access* **2021**, *9*, 11585–11597. [CrossRef]
32. Chen, S.; Xu, S.; Chen, X.; Li, F. Image Denoising Using a Novel Deep Generative Network with Multiple Target Images and Adaptive Termination Condition. *Appl. Sci.* **2021**, *11*, 4803. [CrossRef]
33. Kim, Y.; Soh, J.W.; Park, G.Y.; Cho, N.I. Transfer learning from synthetic to real-noise denoising with adaptive instance normalization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 3482–3492.
34. Ramachandran, P.; Zoph, B.; Le, Q.V. Swish: A Self-Gated Activation Function. *arXiv* **2017**, arXiv:1710.05941.
35. Misra, D. Mish: A self regularized non-monotonic activation function. *arXiv* **2019**, arXiv:1908.08681.
36. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
37. Lim, B.; Son, S.; Kim, H.; Nah, S.; Mu Lee, K. Enhanced deep residual networks for single image super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Honolulu, HI, USA, 21–26 July 2017; pp. 136–144.
38. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
39. Zhang, Y.; Tian, Y.; Kong, Y.; Zhong, B.; Fu, Y. Residual dense network for image super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2472–2481.

40. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015; pp. 1026–1034.
41. Plotz, T.; Roth, S. Benchmarking denoising algorithms with real photographs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1586–1595.
42. Abdelhamed, A.; Lin, S.; Brown, M.S. A High-Quality Denoising Dataset for Smartphone Cameras. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1692–1700. [CrossRef]
43. Tian, C.; Xu, Y.; Li, Z.; Zuo, W.; Fei, L.; Liu, H. Attention-guided CNN for image denoising. *Neural Netw.* **2020**, *124*, 117–129. [CrossRef] [PubMed]
44. Yue, Z.; Yong, H.; Zhao, Q.; Meng, D.; Zhang, L. Variational Denoising Network: Toward Blind Noise Modeling and Removal. In *Advances in Neural Information Processing Systems 32*; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: Vancouver, BC, Canada, 2019; pp. 1690–1701.
45. Zhang, K.; Li, Y.; Zuo, W.; Zhang, L.; Van Gool, L.; Timofte, R. Plug-and-Play Image Restoration with Deep Denoiser Prior. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 6360–6376. [CrossRef] [PubMed]