IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 23, 2019, accepted May 10, 2019, date of publication May 17, 2019, date of current version May 28, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2917537

Adaptively Tuning a Convolutional Neural Network by Gate Process for Image Denoising

YOONSIK KIM[1], JAE WOONG SOH[1], (Student Member, IEEE), AND NAM IK CHO[1], (Senior Member, IEEE)
Department of Electrical and Computer Engineering, INMC, Seoul National University, Seoul 08826, South Korea

Corresponding author: Nam Ik Cho (nicho@snu.ac.kr)

This work was supported by the Ministry of Trade, Industry, and Energy (MOTIE), South Korea, under the "Regional Specialized Industry Development Program (R&D, P0002072)" supervised by the Korea Institute for Advancement of Technology (KIAT) under Grant P0002072.

ABSTRACT The conventional image denoising methods based on the convolutional neural network (CNN) focus on the non-blind training, and hence many networks are required to cope with various noise levels at the test. Although there are blind training methods that deal with multiple noise levels with a single network, their performance gain is generally lower than the non-blind ones, especially at low noise levels. In this paper, we propose a new denoising scheme that controls the feature maps of a single denoising network according to the noise level at the test phase, without changing the network parameters. This is achieved by employing a gating scheme where the feature maps of the denoising network are multiplied with appropriate weights from a gate-weight generating network which is trained along with the denoising network. We train the overall network on a wide range of noise level such that the proposed method can be used for both blind and non-blind cases. The experiments show that the proposed system yields better denoising performance than the other CNN-based methods, especially for the untrained noise levels. Finally, it is shown that the proposed system can manage spatially variant unknown noises and real noises without changing the whole CNN parameters.

INDEX TERMS Denoising, adaptive denoiser, flexible denoiser, convolutional neural network (CNN), noise level estimation, gate-weight generating network.

I. INTRODUCTION

With the development of deep CNN for image classification problems [1]–[3], there have also been many researches to apply the CNN to the image restoration problems [4]–[12]. Especially for image denoising, there have been significant improvements for reducing the real noises as well as the synthetic additive white Gaussian noises [13]–[18]. However, the conventional CNN-based methods have a drawback when they are trained in a non-blind manner, *i.e.*, when they are trained for the specific noise levels because we need to prepare a large number of trained models to cope with various-level or spatially variant noises.

To alleviate the problem of non-blind methods, we may train the network in a blind manner as in DnCNN-B [15] and REDNet [14], which is to train a single network with the images from a wide-range noise level. They show comparable or even better performance than the specific models in some noise range, through the training with the augmented data from similar noise levels. However, the previous blind denoisers have limitations that they do not work well for the complex and low-level noisy images, and also prone to produce blurry outputs. It is because the blind denoising is a more complicated problem than the non-blind, and it seems that the network learns the average noise level when the inputs with wide-range noise level are used for the training. For a more practical case that the noise level is spatially variant, Zhang *et al.* [19] proposed a flexible network which takes noise level maps as the input for learning various noise levels with a single network.

In this paper, we propose a new single-model denoising CNN that replaces multiple networks which are non-blindly

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

trained for the specific noise levels. The proposed system can be used either as a blind or non-blind denoiser, but it is different from the respective conventional methods in several aspects. First, unlike the traditional non-blind denoising methods that use multiple networks, we use a single denoising CNN which works adaptively according to the locally different noise level, without changing the parameters. For this, we adopt the gate scheme where the activation of each feature map is controlled by using gate-weights which are learned along with the CNN parameters. Concretely, in addition to the denoising network, we design and train a small network that generates appropriate gate-weights for the given noise level. Then, the denoising network is tuned to a noise level by scaling the features instead of changing the parameters. Second, unlike the conventional blind denoiser that is trained and used without considering the noise level, we train the combination of denoising and gate-weight generating networks with noise information, and hence we need a noise estimator for the blind scheme. For this, we design a simple noise estimator which produces the (spatially variant) noise level map for the given image. The map is fed to the gate-weight generating network which produces appropriate weights to update feature maps of the denoising network.

In summary, the proposed denoising system consists of a CNN, a noise level estimator, and a gate-weight generating network, which is trained to reduce the spatially variant noises with unknown levels. We believe that our contributions are as follows:

- We propose a single-model CNN denoiser that adaptively works for a wide range of noise level. Moreover, it can be applied to spatially variant noises.
- The adaptation is to scale the feature maps of the CNN for the given noise level where the scaling weights for each noise level are also learned at the training phase. Hence, there are no changes in the parameter values at the test phase, whereas the conventional non-blind methods reload all the parameters or switch to other networks for the change of noise level.
- We also propose a noise level estimation network which requires few parameters.
- Using the proposed noise estimation module, denoiser, and gate-weight generating network, the proposed system can be applied to blind spatially variant noisy images which have a similar property with real noisy images. Hence the proposed method also works well for the real noisy images from cameras or old photos.

A representative example is presented in Fig. 1 which compares the conventional method and our spatially adaptive method.

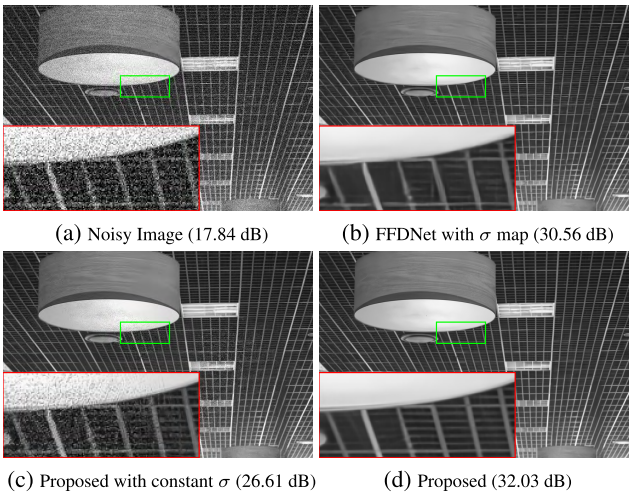
(a) Noisy Image (17.84 dB) (b) FFDNet with σ map (30.56 dB)
(c) Proposed with constant σ (26.61 dB) (d) Proposed (32.03 dB)

FIGURE 1. A noisy image and the denoising results. (a) An image with the spatially variant noise level. Results of (b) FFDNet using the ground truth noise level map, (c) proposed ATDNet with the single noise level (average of noise level over the image), and (d) proposed ATDNet which works adaptively to spatially variant noise. We can find that the proposed system can cope with spatially variant noises by comparing (c) and (d). Moreover, we can see that the proposed denoiser keeps the textures by comparing (b) and (d).

II. PROPOSED SPATIALLY ADAPTIVE DENOISING ARCHITECTURE

A. OVERVIEW OF THE PROPOSED ARCHITECTURE

We develop a CNN-based image denoiser which reduces the spatially variant noise adaptively, without knowing the true noise levels. For this, we need a noise level estimator in addition to the noise removing network named as adaptively tuned denoising network (ATDNet). The overall architecture is shown in Fig. 2, which illustrates that the noise level estimator produces the noise standard deviation at each pixel position, *i.e.*, the noise level map $\boldsymbol{\sigma} \in R^{H\times W}$ where $H \times W$ is the image size. The noise estimator is a simple convolutional network which is trained independently of others. The noise level map is fed to a small network (Network G in the figure) which generates the gate-weight to tune the feature maps of the denoising CNN (Network F in the figure) which consists of gate-residual blocks (Gate-ResBlock).

When training the network, several levels of spatially invariant noises are used for simplifying the training phase, although we are dealing with the spatially variant noise at the test. Specifically, the noise level map at the training phase is $\sigma \times \mathbb{1}^{H\times W}$ for several different σ values, where $\mathbb{1}^{H\times W}$ is the $H \times W$ matrix with all the elements 1.

B. MOTIVATION OF THE PROPOSED ATDNet

Unlike the conventional methods which prepare a CNN for each noise level (non-blind method) or which design a single CNN of which parameters are trained using the images from a wide range of noise level (blind method), we design a single network that can be adapted to the noise level without changing the network parameters. For achieving this, we believe that an easy way is to scale up/down the feature maps of the CNN because they contain enough information for image restoration while each map may contribute differently depending on the noise levels. In other words, the CNN feature maps can be tuned to the noise level by controlling the contribution of each feature map. These motivations and ideas lead us to design a gate scheme, which adds a gate to

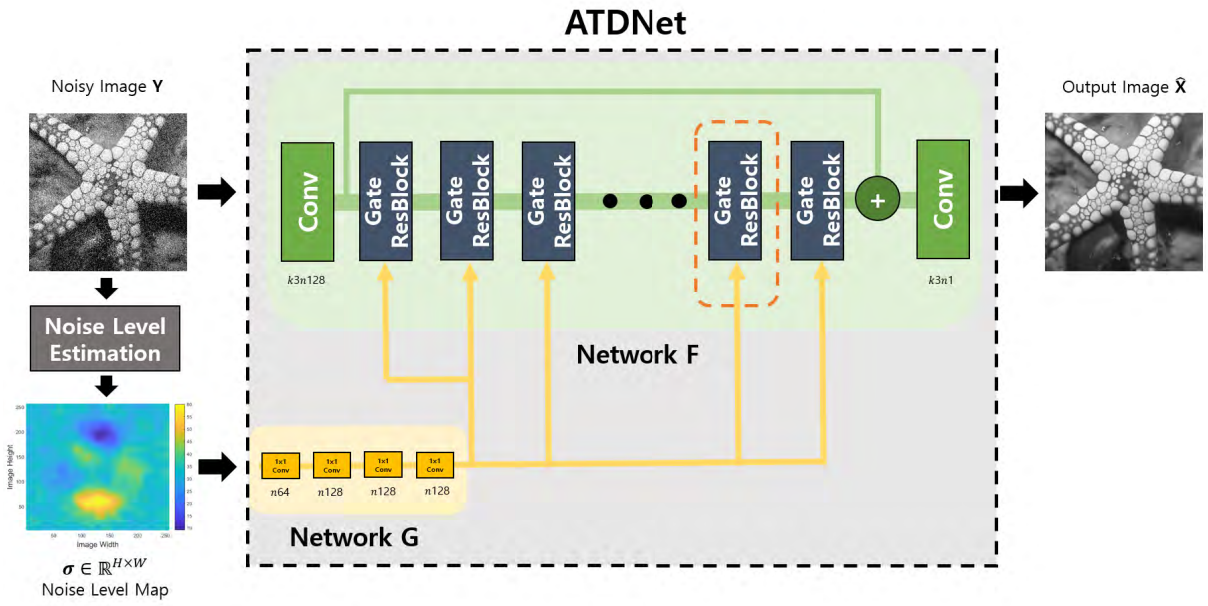

FIGURE 2. The proposed system consists of a noise level estimation module (Noise Level Estimation in the left side) and ATDNet which is composed of denoising network *F* and gate-weight generating network *G* with kernel size (*k*) and the number of features (*n*). The *F* is composed of a gate-residual block (Gate-ResBlock) which is the stack of two 3 × 3 convolutions and a gate process as illustrated in Fig. 3. The number of overall Gate-ResBlocks is 32, and the number of features in each layer is 128. The *F* has skip connection so that the features in the Gate-ResBlock mostly contain the noises and high-frequency components. The *G* is composed of four 1 × 1 convolutions which have 64, 128, 128 and 128 channels respectively. Note that the number of output channels (128) is matched to the number of feature maps in the Gate-ResBlock for the element-wise multiplication of the output from the *G* and the feature maps. The numbers of training parameters for noise level estimation module, network *G*, and *F* are 139 K, 41 K, and 9.587 M respectively.

each residual block of the conventional CNN structure. The role of the gate is to control each map's magnitude at each layer. The gate parameters (weights that are multiplied to the features) are also trained along with the input and noise level so that the feature maps are appropriately adjusted depending on the noise levels.

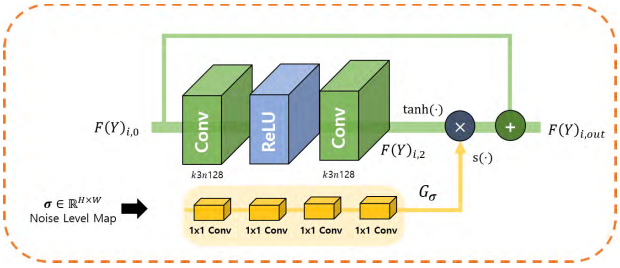

FIGURE 3. The *i*-th Gate-ResBlock with kernel size (*k*) and the number of features (*n*).

C. DENOISING NETWORK F

Inspired from the recent success of residual blocks in the image enhancement tasks [12], [20], we compose denoising network F with the proposed Gate-ResBlock that is illustrated in Fig. 3. F takes a noisy image as an input and also takes the representations of noise level map, which are generated from Network G, as a conditional input. Then, F can be flexibly tuned to a wide range of noise level and generates a denoised image by reducing noises with a single network.

D. GATE-WEIGHT GENERATING NETWORK G

The gate-weight generating network G takes the noise level map ($H \times W$ matrix where each element represents standard deviation at the pixel position, scaled to [0, 1]) as the input, and outputs $H \times W \times 128$ gate-weights that are element-wise multiplied to the $H \times W \times 128$ feature maps of Gate-ResBlock as the "gate process" illustrated in Fig. 3. Precisely, the network G is composed of four 1×1 convolution filters with 64, 128, 128 and 128 feature maps respectively, so that the output size of G is $H \times W \times 128$ accordingly. Then, the gate weights determine the amount of pixel-wise feature map changes. Precisely, our Gate-ResBlock updates the input features by gating them according to the noise level with the residual connection.

Formally, the output of G is a function of noise variance $\sigma \in R^{H \times W}$ which is represented as $G_\sigma \in R^{H \times W \times 128}$. Meanwhile, let $F(Y)_{i,j} \in R^{H \times W \times 128}$ be the outputs of the j-th convolution in the i-th Gate-ResBlock as denoted in Fig. 3. Then the operation at each Gate-ResBlock in Fig. 3 can be written as

$$F(Y)_{i,out} = F(Y)_{i,0} + \alpha \tanh(F(Y)_{i,2} \circ s(G_\sigma)) \qquad (1)$$

where $s(\cdot)$, $\circ$, and α are sigmoid function, element-wise product, and the update scaling parameter (10^{-1}) respectively. The proposed Gate-ResBlock makes the Network F to be adaptive to the change of the noise level. In another point of view, it acts as feature attention [21], [22] in that it recalibrates the feature maps according to the noise level.

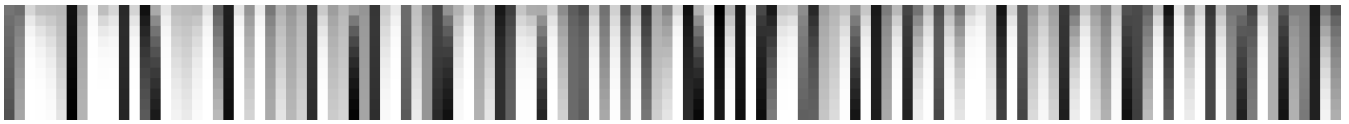

FIGURE 4. Visualization of output of G. Each row represents the 1×128 vector $G_\sigma(m, n, :)$ at a pixel position (m, n), for a particular σ, where σ ranges from 10 to 60 with the step size 5. For example, the first row is the vector $G_{10}(m, n, :)$, the second row $G_{15}(m, n, :)$, and the last row $G_{60}(m, n, :)$.

TABLE 1. The proposed noise level estimation module.

Type	Kernel / Stride Size	Output Size
Conv1, 2	$3 \times 3 / 1 \times 1$	$H \times W \times 32$
Max Pool1	$2 \times 2 / 2 \times 2$	$\frac{H}{2} \times \frac{W}{2} \times 32$
Conv3, 4	$3 \times 3 / 1 \times 1$	$\frac{H}{2} \times \frac{W}{2} \times 64$
Max Pool2	$2 \times 2 / 2 \times 2$	$\frac{H}{4} \times \frac{W}{4} \times 64$
Conv5	$3 \times 3 / 1 \times 1$	$\frac{H}{4} \times \frac{W}{4} \times 128$
Avgerage Pool1	$2 \times 2 / 2 \times 2$	$\frac{H}{8} \times \frac{W}{8} \times 128$
Conv6	$1 \times 1 / 1 \times 1$	$\frac{H}{8} \times \frac{W}{8} \times 1$
Average Pool2	Global Pool	$1 \times 1 \times 1$

E. NOISE LEVEL ESTIMATION MODULE

The proposed noise level estimation module is composed of a few convolutions as described in Table 1. It can be considered that the last layer (Conv6) outputs the block-wise noise level, where the block size is 8×8. If we wish to denoise the image with (assumed) spatially invariant noise, then we average the values from the output of Conv6, and the average noise level σ is fed to the network F in the form of $\sigma \times \mathbb{1}^{H \times W}$. Otherwise, if we wish to deal with the pixel-wise noise level for the more practical case of spatially variant noise, the $\frac{H}{8} \times \frac{W}{8}$ matrix (output of Conv6) is linearly interpolated to be the $H \times W$ noise level map which is fed to the network F.

F. TRAINING

We train the proposed system using the noisy patches $\mathbf{y}_i$ and the corresponding clean patches $\mathbf{x}_i$, with the noise standard deviation σ_i in $[10, 60]$ with the step size 5. The patch size is empirically determined as 80×80 and the number of batch size as 16. The overall loss function for the training is defined as

$$\mathbf{L}(\Theta) = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{x}_i - F(\mathbf{y}_i, \hat{\boldsymbol{\sigma}}_i; \Theta)\|_2^2 \tag{2}$$

$$+ \frac{1}{N} \sum_{i=1}^{N} \|\boldsymbol{\sigma}_i - E(\mathbf{y}_i; \Theta)\|_2^2 \tag{3}$$

where $E(\cdot)$ and $F(\cdot)$ are the outputs of noise level estimation module and ATDNet respectively. When the baseline network for the single-level denoising converges within K iterations, the proposed method usually converges within $2K$. We can say that this is another advantage of our approach over the conventional non-blind method which needs KM iterations when preparing the networks for M different noise levels.

To be precise with the training environments, we use the ADAM optimizer and set the initial learning rate to 5×10^{-4} which decays to 10^{-4} after 10 epochs. All the experiments are carried out in a PC with Intel (R) Core(TM) i7-6850K CPU 3.60GHz and the Nvidia Titan X GPU which has 12GB memory, and the proposed network is implemented with Tensorflow.

III. DISCUSSION

In this section, we discuss the effect of the proposed method by visualizing the gate-weight values from G for several cases and also visualizing the gating process in (1).

A. VISUALIZATION OF WEIGHTS FROM G

When there is no G in Fig. 2, or when all the output of G is 1 regardless of its input σ, then the network is the same as the conventional denoiser that uses the baseline network F. Hence, we first show in Fig. 4 that the G outputs quite widely varying values for different noise level. Each row in the figure is a 1×128 vector extracted from G for the noise level σ, and we show 11 vectors from $\sigma = 10$ to 60 with the step size 5. The sigmoid function $s(\cdot)$ is designed such that an output node of G can have the values in $[0, 1]$, which is scaled to $[0, 255]$ for the visualization. Note that the output of G, *i.e.*, G_σ is an $H \times W \times 128$ tensor, and the rows in Fig. 4 are the instances of $G_\sigma(m, n, :)$ for some pixel position (m, n) for $\sigma = 10 : 5 : 60$.

From the figure, we can see that the element of G_σ changes almost monotonically as σ changes (the change into vertical direction). On the other hand, the changes into the horizontal direction (changes in output node values that are multiplied to the feature maps) are quite large. This means that the contribution of each feature map is quite different, while the change of each feature's contribution according to the σ is monotonic.

B. VISUALIZATION OF FEATURE MAP UPDATE

The residual network is to learn the difference between the input and output, which is mostly noise in our problem. Hence, the final Gate-ResBlock needs to produce only noise component when successfully converged. In this subsection, we visualize that our main process in (1) helps to reduce the features that have image structure so that the Gate-ResBlock gives more desirable outputs. Specifically, Fig. 5 shows the actual feature maps in our network, and the visualization of (1). We can see that there are some feature maps which have strong image structures among $F(Y)_{i,2}$ (especially the one marked with red boundary), and our scheme multiplies

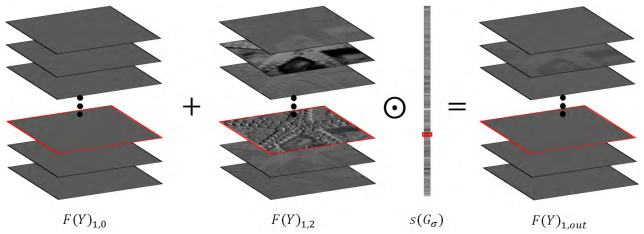

FIGURE 5. Visualization of feature maps generated from F. It shows that $s(G_\sigma)$ gates $F(Y)_{1,2}$, *i.e.,* $s(G_\sigma)$ scales up/down the feature maps. Red boxes are the representative examples of feature map update.

a small weight to such features so that a noise-like image is obtained at the output. The variances of feature maps of $F(Y)_{i,0}$, $F(Y)_{i,2}$, and $F(Y)_{i,out}$ are 0.027, 4.678, and 0.021 respectively, which also shows that our update scheme has the role of regularizing the feature maps.

IV. DATASETS AND EXPERIMENTS

We conduct experiments on gray and color images corrupted by synthetic and real noises.

A. DATASETS AND DEGRADATION OF IMAGE

The datasets used for the gray AWGN evaluation are Set12 [15], BSD68 [23], and Urban100 [24]. The first two sets are used in many conventional works, and the Urban100 is recently popularly used as it contains more textured images such as the facades and ceilings of modern buildings. We train the network with $\sigma \in [10, 60]$, but show the test results just on $\sigma = 15, 25, 30$, and 50, because the compared algorithms in [14] and [15] show the results for these values. For the real noise experiments, we use the RNI6 [25]. For the color AWGN evaluation, CBSD68 [23], Kodak24 [26], McMaster [27] and CUrban100 [24] are used for the evaluation.

B. BASELINE AND COMPARED METHODS

We compare our methods with recent denoising networks, and also with the baseline network (network F without our gate G) to show the gain of our gating scheme over the plain use of ResBlocks. For comparing both blind and non-blind cases, we consider three different use of baselines depending on the use of noise information:

- Baseline-S: The network is trained to specific degradation levels.
- Baseline-B: The network is trained with the images from all over the degradation level.
- Baseline-C: The network is trained with the images from all over the degradation level taking the concatenated noise level maps with images as an input, which is employed in [6] and [19].

With these prepared networks, we conduct blind and non-blind experiments separately. Specifically, the proposed ATDNet is compared with BM3D [28], TNRD [29], REDNet [14], DnCNN-S [15], MemNet [30], UNLNet [31], FFDNet [19], Baseline-S and Baseline-C for non-blind test, and the proposed ATDNet-EST is compared with DnCNN-B and Baseline-B for the blind test. The ATDNet, Baseline-S, Baseline-C, and Baseline-B are trained with BSD400 [32] set, which has a similar number of images used for the training of REDNet and DnCNN. Also, considering the capacity of the proposed architecture, we also train our network with more datasets (BSD400 + DIV2K 800 images [33]) which amounts to the similar number of training patches as used for the FFDNet, and this network will be called ATDNetW (Wider range of training data).

C. EXPERIMENTS

1) EXPERIMENTS ON AWGN REMOVAL

Table 2 provides the performance of the proposed noise-level estimation module. We can see that the proposed estimation module accurately estimate the noise level with small error and standard deviation. As a result, the ATDNet-EST can have comparable performance to ATDNet although it does not have noise level information.

TABLE 2. Mean and standard deviation of the proposed noise level estimation module.

σ	Set12	BSD68	Urban100
15	14.92 ± 0.28	14.93 ± 0.31	15.06 ± 0.49
25	24.92 ± 0.28	25.03 ± 0.17	25.08 ± 0.39
30	29.92 ± 0.28	30.06 ± 0.24	30.03 ± 0.41
50	49.17 ± 0.37	49.47 ± 0.61	49.51 ± 0.61

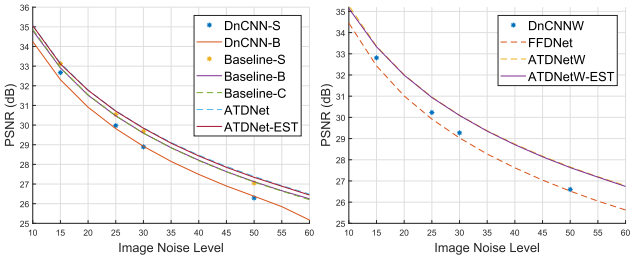

FIGURE 6. Average PSNR obtained by conventional and the proposed methods for some noise levels. Non-blind methods are represented with dots or dashed lines, and blind methods are represented with solid lines. The validation is evaluated on Urban100.

Tables 3 and 4 present the average PSNR of above-stated methods at some noise levels. We can observe that the blind training methods (both DnCNN-B and Baseline-B) yield about 0.2dB lower performance than the respective specific models at low noise levels. In contrast, ATDNet shows comparable performance to the Baseline-S at $\sigma = 15$, and even gets better performance at other noise levels. Moreover, the ATDNetW achieves the best performance for all datasets and noise levels. We can also find from Fig. 6 that our method shows almost the same results for blind and non-blind tests. Specifically, the figure is the plot of average PSNR vs. noise level, where we can see that proposed methods achieve better or comparable performance than the others.

TABLE 3. Average PSNR of the restored images, where the inputs are corrupted by AWGN with $\sigma = 15, 25, 30$, and 50, for the images from Set12, BSD68, and Urban100 datasets. The compared and proposed networks are trained with BSD400. The left part is the non-blind test, and the right is blind (red: The best result, blue: The second best). Since the proposed noise level estimation module provides quite accurate estimates, ATDNet-EST can have same results with ATDNet.

Set12												
Sigma	BM3D	TNRD	REDNet	DnCNN-S	MemNet	UNLNet	Baseline-S	Baseline-C	ATDNet	DnCNN-B	Baseline-B	ATDNet-EST
15	32.38	32.50	-	32.86	-	32.67	33.04	32.91	33.03	32.68	32.92	33.03
25	29.95	30.04	-	30.44	-	30.25	30.67	30.65	30.77	30.36	30.65	30.73
30	29.15	-	29.62	29.52	29.64	29.36	29.89	29.84	29.95	29.53	29.82	29.92
50	26.70	26.78	27.26	27.18	27.40	27.04	27.59	27.53	27.63	27.21	27.57	27.60
BSD68												
Sigma	BM3D	TNRD	REDNet	DnCNN-S	MemNet	UNLNet	Baseline-S	Baseline-C	ATDNet	DnCNN-B	Baseline-B	ATDNet-EST
15	31.07	31.42	-	31.73	-	31.47	31.82	31.76	31.84	31.61	31.77	31.82
25	28.56	28.91	-	29.23	-	28.98	29.34	29.33	29.39	29.17	29.32	29.40
30	27.75	-	28.50	28.36	28.45	28.14	28.56	28.52	28.60	28.35	28.51	28.58
50	25.62	25.96	26.37	26.23	26.37	26.04	26.42	26.43	26.50	26.23	26.42	26.45
Urban100												
Sigma	BM3D	TNRD	REDNet	DnCNN-S	MemNet	UNLNet	Baseline-S	Baseline-C	ATDNet	DnCNN-B	Baseline-B	ATDNet-EST
15	32.34	31.96	-	32.67	-	32.44	33.13	32.92	33.15	32.31	32.91	33.11
25	29.70	29.21	-	29.97	-	29.77	30.54	30.47	30.72	29.82	30.47	30.71
30	28.75	-	28.98	28.89	29.11	28.75	29.68	29.59	29.86	28.92	29.59	29.84
50	25.95	25.59	26.34	26.29	26.65	26.12	27.05	27.10	27.39	26.38	27.12	27.34

TABLE 4. Average PSNR of the restored images, where the inputs are corrupted by AWGN with $\sigma = 15, 25, 30$, and 50, for the images from Set12, BSD68, and Urban100 datasets. The compared and proposed networks are trained with BSD400 and more datasets. The left part is the non-blind test, and the right is blind (red: The best result, blue: The second best).

Set12				
Sigma	FFDNet	DnCNNW-S	ATDNetW	ATDNetW-EST
15	32.75	32.87	33.06	33.07
25	30.43	30.49	30.79	30.77
30	29.61	29.67	30.00	29.98
50	27.32	27.28	27.70	27.63
BSD68				
Sigma	FFDNet	DnCNNW-S	ATDNetW	ATDNetW-EST
15	31.63	31.78	31.83	31.82
25	29.19	29.31	29.39	29.39
30	28.39	28.49	28.58	28.57
50	26.29	26.38	26.49	26.45
Urban100				
Sigma	FFDNet	DnCNNW-S	ATDNetW	ATDNetW-EST
15	32.43	32.81	33.36	33.33
25	29.92	30.23	30.96	30.94
30	29.03	29.27	30.10	30.09
50	26.52	26.60	27.66	27.63

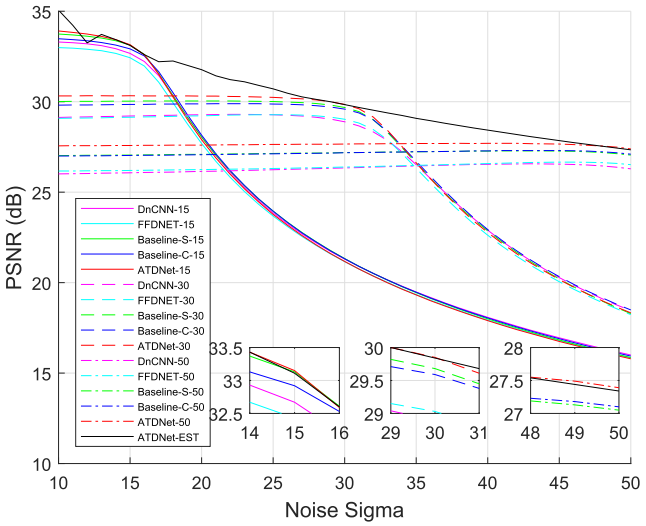

FIGURE 7. Sensitivity curves that represent how the model-N (N = 15, 30, 50) works when the noise level is different from the trained one (or estimated one). Some parts are magnified at the bottom to show the difference more clearly.

We also demonstrate the robustness of our method to the noise level estimation error in Fig. 7 which shows the performance when the noise level is mismatched with the real noise level. Precisely, the figure provides how the applied (or trained) models such as FFDNet-S-N, Baseline-C-N, and ATDNet-S-N (or DnCNN-S-N and Baseline-S-N) work when the real noise level is different from the applied (or trained) one, *i.e.,* when $\sigma \neq N$. As shown in the figure, the proposed ATDNet is more robust than the others in that a small mismatch in the noise level does not result in a large performance degradation. We can also find that the ATDNet-EST works robustly for the fine-scale noise levels, *i.e.,* for the noise level with the steps of 1 where ATDNet-EST is trained for noise level in [10, 60] with the steps of 5. From these quantitative results, we can conclude that the bundles of specifically trained models can be replaced by our single network for both non-blind and blind denoising.

A distinct difference between the blind (-B) and non-blind methods is the blurriness of the output. Since the non-blind model is trained only for a specific noise level, it can remove noise while preserving the textures, whereas the blind model learns the averaged properties so that the results are somewhat blurry. An example is shown in Fig. 8 where the noisy input image is from the Urban100 dataset that has many textures

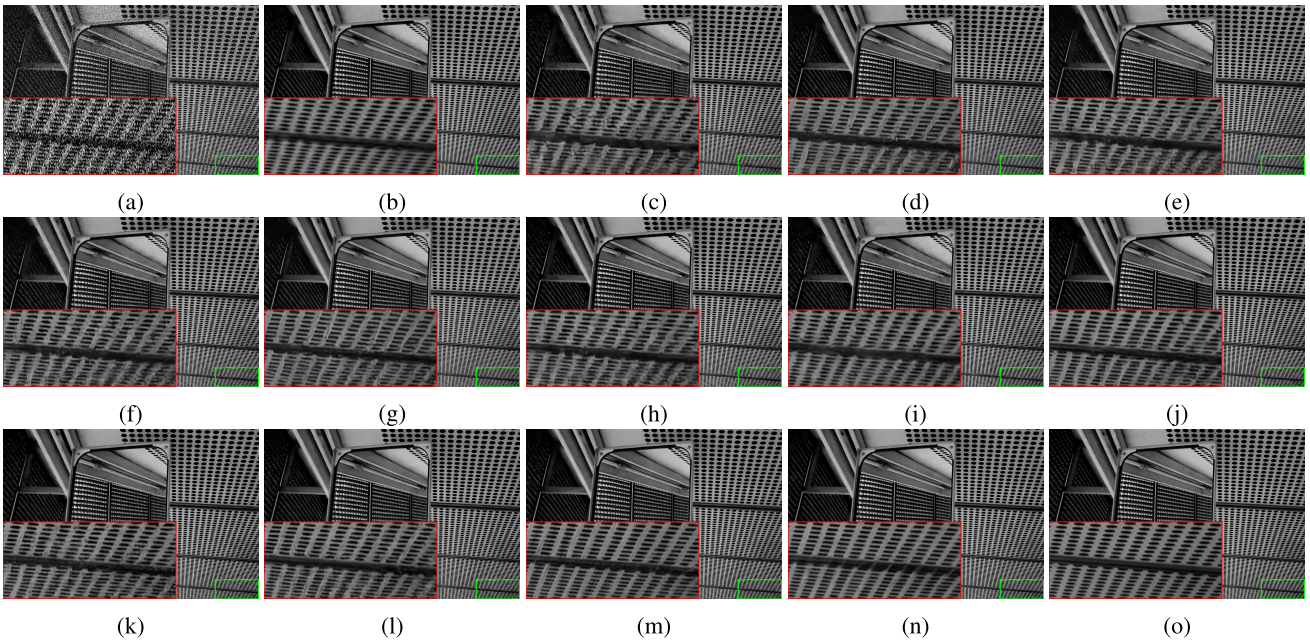

FIGURE 8. The 4th image from the Urban100 dataset corrupted by AWGN with $\sigma = 50$, and the comparison of various methods. (a) Noisy image. (b) BM3D/25.86 dB. (c) TNRD/24.84 dB. (d) REDNet/25.67 dB. (e) DnCNN-S/25.67 dB. (f) DnCNN-B/25.79 dB. (g) MemNet/25.74 dB. (h) FFDNet/25.96 dB. (i) UNLNet/25.79 dB. (j) Baseline-S/26.66 dB. (k) Baseline-B/26.66 dB. (l) Baseline-C/26.68 dB. (m) ATDNet/27.01 dB. (n) ATDNetW/27.54 dB. (o) Ground-truth.

and edges. From the figures, we can observe that ATDNet and ATDNetW yield clear images compared to the baselines and conventional methods around the edges.

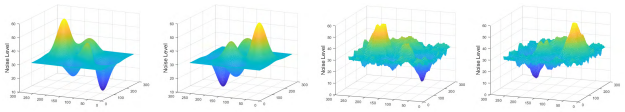

FIGURE 9. True noise level maps and their estimations by the proposed method. (a) Noise level maps. (b) Estimated noise level maps.

TABLE 5. Average PSNR of the restored images, where the inputs are corrupted by spatially variant AWGN with average $\sigma = 35$ for the images from Set12, BSD68, and Urban100 datasets. The left upper part is the non-blind test, and the lower is blind (red: The best result, blue: The second best).

Dataset	Set12	BSD68	Urban100
FFDNet	29.08	27.88	28.43
Baseline-C	29.11	27.89	29.21
ATDNet	29.43	28.09	29.26
ATDNetW	29.46	28.08	29.52
Baseline-B	28.59	27.58	28.99
ATDNet-EST	29.28	28.01	29.11
ATDNetW-EST	29.32	28.01	29.38

TABLE 6. Average PSNR of the restored color images, where the inputs are corrupted by AWGN with $\sigma = 15, 25, 30$, and 50. The Left part is the non-blind test and right is blind (red: The best result, blue: The second best). Since the proposed noise level estimation module provides quite accurate estimates in this case, CATDNetW-EST can have almost the same results with CATDNetW.

CBSD68					
Sigma	CFFDNet	CUNLNet	CATDNetW	CDnCNN-B	CATDNetW-EST
15	33.88	33.87	34.08	33.89	34.08
25	31.15	31.21	31.48	31.23	31.48
30	30.21	30.31	30.60	30.32	30.60
50	27.85	27.96	28.30	27.92	28.30
Kodak24					
Sigma	CFFDNet	CUNLNet	CATDNetW	CDnCNN-B	CATDNetW-EST
15	34.63	34.64	34.99	34.48	35.00
25	32.08	32.13	32.58	32.03	32.58
30	31.18	31.28	31.75	31.18	31.71
50	28.86	28.98	29.49	28.84	29.49
McMaster					
Sigma	CFFDNet	CUNLNet	CATDNetW	CDnCNN-B	CATDNetW-EST
15	34.33	34.66	35.12	33.44	35.09
25	32.02	32.35	32.87	31.51	32.87
30	31.00	31.52	32.06	30.78	32.02
50	28.80	29.18	29.78	28.61	29.74
CUrban100					
Sigma	CFFDNet	CUNLNet	CATDNetW	CDnCNN-B	CATDNetW-EST
15	33.97	33.83	34.56	32.98	34.53
25	31.50	31.40	32.32	30.81	32.30
30	30.41	30.53	31.52	29.99	31.48
50	27.95	28.05	29.21	27.59	29.19

2) EXPERIMENTS ON SPATIALLY VARIANT NOISE

We also apply our methods, baselines, and FFDNet to spatially variant noisy images. We generate noise level maps as in [19], which are visualized in Fig. 9(a). For these spatially variant noises, Table 5 presents the average PSNR of FFDNet and our algorithms. We can observe that the proposed ATDNet and ATDNetW yield larger gain than the FFDNet and Baseline-C. Moreover, the ATDNet-EST and ATDNetW-EST also show comparable performance to non-blind methods

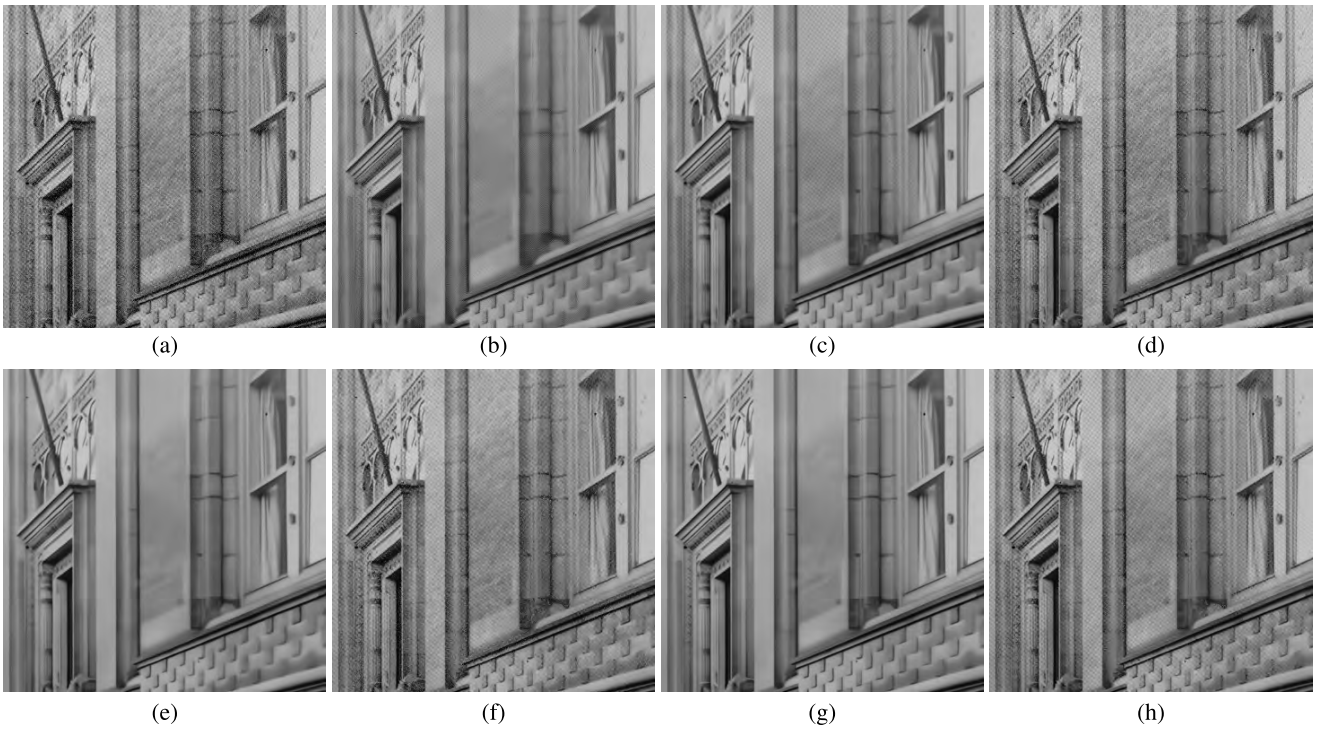

FIGURE 10. The real noisy image 'Building,' and the comparison of the results by various methods with the sigma 20 (estimted in [19]) for ATDNetW, or with the estimated spatially variant noise map for ATDNetW-EST. (a) Noisy image. (b) Noise clinic. (c) BM3D. (d) DnCNN-B. (e) FFDNet. (f) Baseline-B. (g) ATDNetW. (h) ATDNetW-EST.

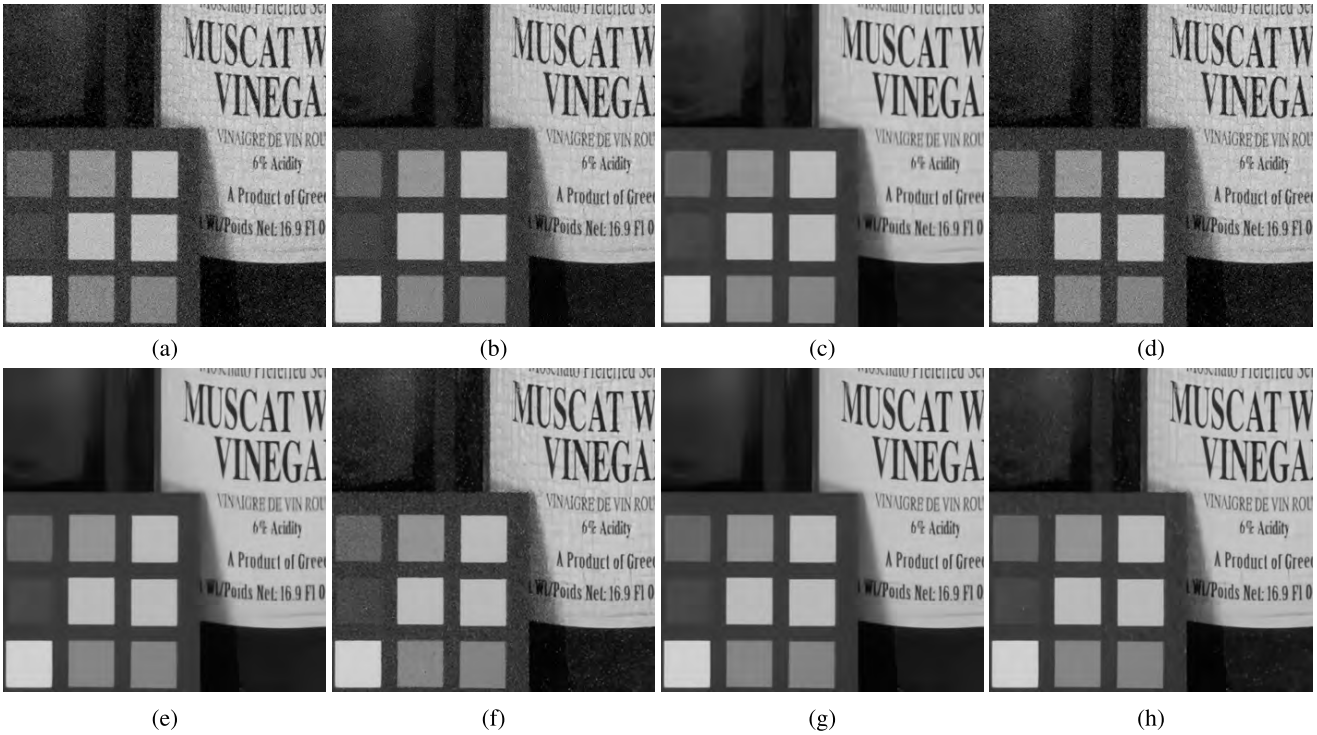

FIGURE 11. The real noisy image 'Vineger,' and the comparison of the results by various methods with the sigma 20 (estimted in [19]) for ATDNetW, or with the estimated spatially variant noise map for ATDNetW-EST. (a) Noisy image. (b) Noise clinic. (c) BM3D. (d) DnCNN-B. (e) FFDNet. (f) Baseline-B. (g) ATDNetW. (h) ATDNetW-EST.

and better performance than Baseline-B, because our noise level estimation network provides quite accurate estimates as shown in Fig. 9(b).

3) EXPERIMENTS ON REAL NOISY IMAGES

From previous experiments, we can observe that the noise level estimator provides accurate results, and the denoiser

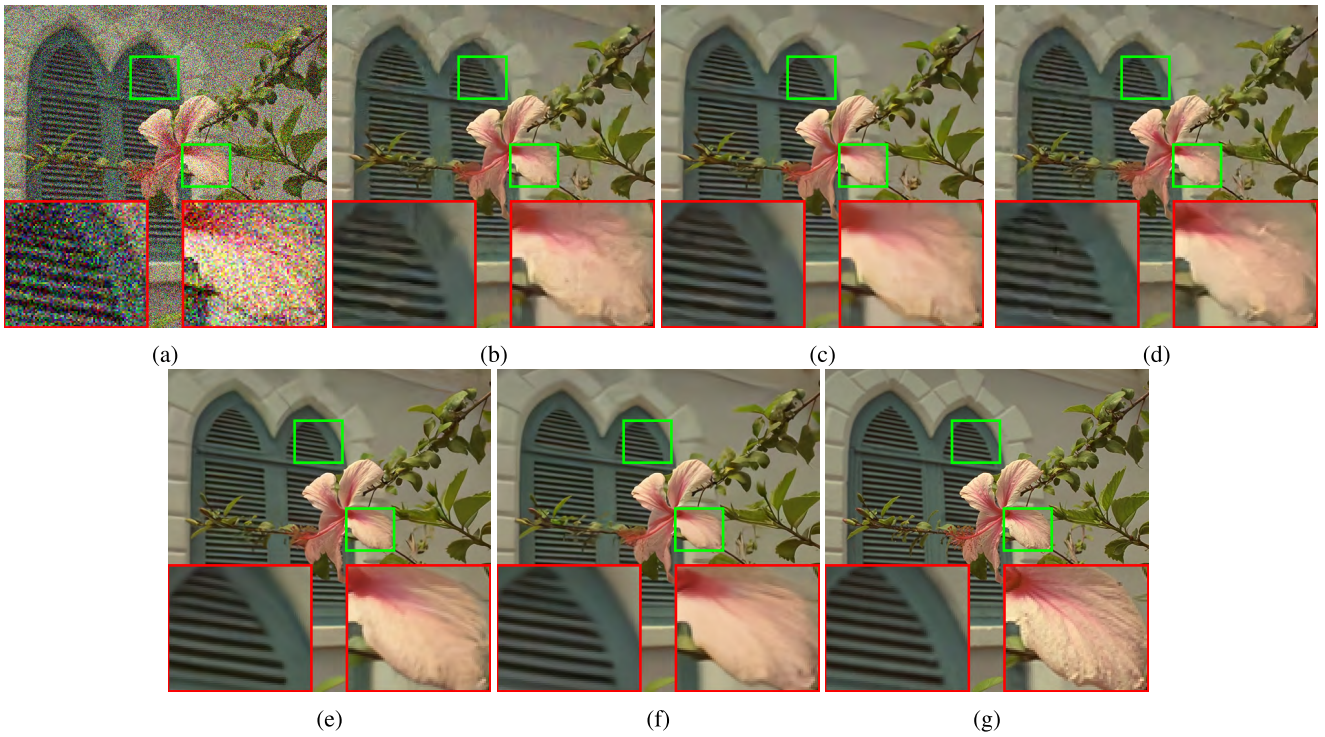

FIGURE 12. The 7th image from the Kodak24 dataset corrupted by AWGN with $\sigma = 50$, and the comparison of the denoising results. (a) Noisy image. (b) CDnCNN-B/ 30.11 dB. (c) CFFDNet/30.54 dB. (d) CUNLNet/30.23 dB. (e) CATDNetW/31.27 dB. (f) CATDNetW-EST/31.23 dB. (g) Ground-truth.

can cope with a wide range of spatially variant noise using a single network. Hence, we apply the proposed scheme to reduce the real noises which are usually spatially variant with unknown level and unknown statistics. We present the results of Baseline-B, ATDNetW and ATDNetW-EST with Noise Clinic [25], BM3D [28], DnCNN-B [15], and FFDNet [19] in Figs. 10 and 11. For the ATDNetW, we use previously estimated noise level [19], and the single noise value is applied to the whole image. For the ATDNetW-EST, we use spatially variant noise level map estimation, and the estimated map is applied. It can be seen that the ATDNetW removes the real noise very well, but it also removes some textures because it applies the single noise level to the whole image. On the other hand, the ATDNetW-EST removes the real noise while preserving textures.

4) EXPERIMENTS ON COLOR NOISE REMOVAL

We also validate the proposed method on noisy color images in this subsection. The color denoiser is trained in the RGB color space and the results are compared in Table 6, where it can be seen that the proposed systems yield larger gains than the others. The resulting images are presented in Fig. 12, where we can also observe that the proposed methods successfully reduce the noise.

V. CONCLUSION

We have proposed a new blind denoising framework that adjusts the CNN's feature maps for the specific noise levels. Unlike the conventional methods that need to prepare as many sets of parameters as pre-defined noise levels, the proposed method enables a single network to work for various levels. The proposed scheme is to add a simple network to the conventional CNN, which takes the noise level as the input and generates scaling weights that are multiplied to the feature maps at each layer. The additional network is so simple that it can be implemented as a look-up table at the test phase. Hence, the computational complexity of the proposed scheme is almost the same as the baseline CNN. We have tested our method and compared with the state-of-the-art methods in image denoising with lots of validations. The results show that the proposed method achieves comparable or better performance than the others, and the blind scheme sometimes works even better than the non-blind one. Our codes and more result images are available at https://github.com/terryoo/ATDNet.

REFERENCES

[1] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556

[2] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[4] 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.*, Jun. 2012, pp. 2392–2399.

[5] L. Xu, J. S. Ren, C. Liu, and J. Jia, "Deep convolutional neural network for image deconvolution," in *Proc. Adv. Neural. Inf. Process. Syst.*, Dec. 2014, pp. 1790–1798.

[6] 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.*, Jul. 2017, pp. 2808–2817.
[7] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 295–307, Feb. 2016. doi: 10.1109/TPAMI.2015.2439281.
[8] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1646–1654.
[9] C. Dong, Y. Deng, C. C. Loy, and X. Tang, "Compression artifacts reduction by a deep convolutional network," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 576–584.
[10] J. Guo and H. Chao, "Building dual-domain representations for compression artifacts reduction," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, Oct. 2016, pp. 628–644.
[11] Y. Tai, J. Yang, and X. Liu, "Image super-resolution via deep recursive residual network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 2790–2798.
[12] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshop*, Jul. 2017, pp. 136–144.
[13] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Proc. Adv. Neural. Inf. Process. Syst.*, Dec. 2012, pp. 341–349.
[14] X. Mao, C. Shen, and Y.-B. Yang, "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections," in *Proc. Adv. Neural. Inf. Process. Syst.*, Dec. 2016, pp. 2802–2810.
[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] 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.*, Jun. 2018, pp. 3155–3164.
[17] B. Mildenhall, J. T. Barron, J. Chen, D. Sharlet, R. Ng, and R. Carroll, "Burst denoising with kernel prediction networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2502–2510.
[18] A. Abdelhamed, S. Lin, and M. S. Brown, "A high-quality denoising dataset for smartphone cameras," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1692–1700.
[19] 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, 2018.
[20] C. Ledig *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4681–4690.
[21] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.
[22] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, Sep. 2018, pp. 286–301.
[23] S. Roth and M. J. Black, "Fields of experts," *Int. J. Comput. Vis.*, vol. 82, no. 2, pp. 205–229, Apr. 2009.
[24] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5197–5206.
[25] M. Lebrun, M. Colom, and J.-M. Morel, "The noise clinic: A blind image denoising algorithm," *Image Process. Line*, vol. 5, pp. 1–54, Jan. 2015. [Online]. Available: http://demo.ipol.im/demo/125/
[26] R. Franzén. (1999). *Kodak Lossless True Color Image Suite*. [Online]. Available: http://r0k.us/graphics/kodak
[27] L. Zhang, X. Wu, A. Buades, and X. Li, "Color demosaicking by local directional interpolation and nonlocal adaptive thresholding," *J. Electron. Imag.*, vol. 20, no. 2, Apr. 2011, Art. no. 023016.
[28] 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.
[29] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1256–1272, Jun. 2016.
[30] Y. Tai, J. Yang, X. Liu, and C. Xu, "MemNet: A persistent memory network for image restoration," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 4539–4547.
[31] S. Lefkimmiatis, "Universal denoising networks : A novel CNN architecture for image denoising," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3204–3213.
[32] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. IEEE Int. Conf. Comput. Vis.*, Jul. 2001, pp. 416–423.
[33] R. Timofte *et al.*, "NTIRE 2017 challenge on single image super-resolution: Methods and results," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshop*, Jul. 2017, pp. 1110–1121.