IEICE TRANS. INF. & SYST., VOL.E102–D, NO.8 AUGUST 2019

LETTER

Image Denoiser Using Convolutional Neural Network with Deconvolution and Modified Residual Network

Soo-Yeon SHIN[†], ***Member***, **Dong-Myung KIM**[†], ***and*** **Jae-Won SUH**[†a)], ***Nonmembers***

SUMMARY Due to improvements in hardware and software performance, deep learning algorithms have been used in many areas and have shown good results. In this paper, we propose a noise reduction framework based on a convolutional neural network (CNN) with deconvolution and a modified residual network (ResNet) to remove image noise. Simulation results show that the proposed algorithm is superior to the conventional noise eliminator in subjective and objective performance analyses.

key words: *image, denoise, deep learning, CNN*

1. Introduction

High interest in deep learning algorithms has led to major changes in research methods in a variety of research areas, including classical research areas such as image noise reduction, and the results are excellent. Image denoising is performed to recover a clean image from a noisy image that has additive white Gaussian noise (AWGN).

Extensive researches based on prior image knowledge have been performed over the past several decades, and excellent image noise reduction methods have been developed, such as BM3D [1] and WNNM [2]. However, these approaches have the disadvantage that important features can be weakened because both noise and key features in the image are equally subjected to the same filtering operation. To solve this problem, some noise reduction methods based on the partial differnetial equations (PDE) have been introduced [3], [4]. These studies show positive result for edge sharpness, but PDE models tend to cause the result image look blocky. These blocky effects are visually unpleasant and may cause errors in computer vision system, such as the boundaries of block are incorrectly recognized as a feature edge. On the other hand, the recently studied image denoising methods based on the deep learning algorithm [5]–[7] can be flexibly applied to various situations because it uses pre-trained results and has produced excellent results. In particular, the DnCNN [7] model significantly improves visual performance in certain Gaussian noise patterns because it learns the actual noise pattern to predict the noise, but it is vulnerable to a high level of noise signal. In addition, it takes a long time to learn and to run because it consists of deep convolutional layers that use the full resolution features.

Manuscript received August 17, 2018.
Manuscript revised March 28, 2019.
Manuscript publicized May 14, 2019.
[†]The authors are with Chungbuk National University, Korea.
a) E-mail: sjwon@cbnu.ac.kr
DOI: 10.1587/transinf.2018EDL8175

In this paper, we propose a new framework based on CNN to overcome these problems. In order to reduce the computational complexity, the proposed model uses low-resolution layers that can reduce the number of feature pixels to be processed in the middle of the network. Also, to avoid gradient loss problems that can occur during iterative learning in deep line models, we use long and short connections.

2. Proposed DRNet Framework

The proposed framework is a deep convolutional neural network with deconvolution and modified residual network (DRNet), as shown in Fig. 1. Note that n_i means input noisy image and $DN(n_i)$ is denoised image. The numbers in the figure are different from the real ones, but they represent the result of each processing. By the two stride convolution, the size of the image is cut in half. It is restored to its original size by two stride deconvolution. After two stride convolution, we made a short connection for every two convolution layers to perform an identity mapping, and we made a long connection to compensate for the down sampled operation.

3. Training

The proposed DRNet is a deep CNN model with 17 convolutional layers to generates a denoised image $DN(n_i)$. The details of our DRNet are listed in Table 1. W and H are the width and height of the input image, F is the number of

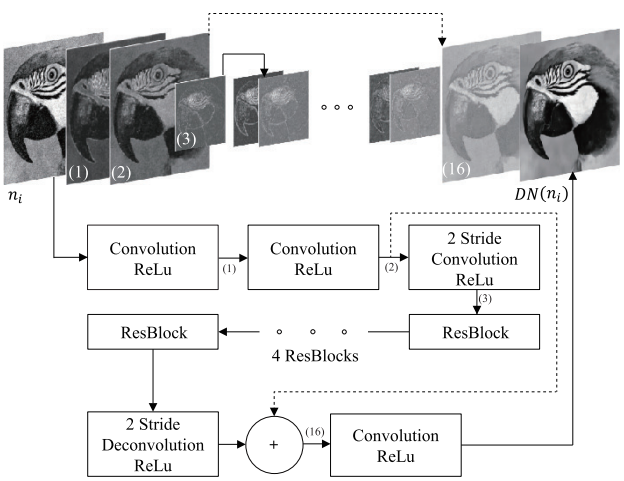

Fig. 1 Architecture of proposed DRNet model layers.

Table 1 Summary of proposed DRNet model architecture.

	Layer Description	Ouput Dim
	input image	H × W × 1
1	5 × 5 convolution, 128 features	H × W × 2F
2	3 × 3 convolution, 64 features	H × W × F
3	3×3 convolution with 2 stride, 64 features	$\frac{1}{2}$ H × $\frac{1}{2}$ W × F
4–15	3 × 3 convolution, 64 features Short connection between layer 3 and layer 5 (ResBlock) 6 consecutive ResBlocks	$\frac{1}{2}$ H × $\frac{1}{2}$ W × F
16	3 × 3 deconvolution with 2 stride, 64 feature Long connection betweeen layer 2 and layer 16	H × W × F
17	3 × 3 convolution	H × W × 1

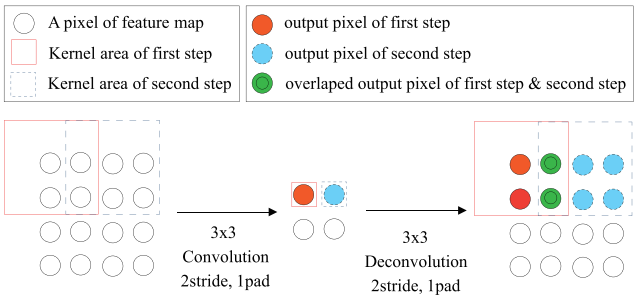

Fig. 2 The kernel area of convolution and deconvolution with 2 stride.

features specified in that layer, and its value is 64.

In the proposed DRNet, the first layer creates 128 features, which are the key features of the original image. We also use the 5 × 5 kernel to extract features over a large area. The convolutional layers of our model are activated by the Rectifier Linear Unit (ReLu). In the second layer, input of 128 features is reduced to 64 features using the 3 × 3 kernel.

In order to effectively reduce training time, the 3^{rd} layer uses the two stride 3 × 3 convolution. Two stride convolution means that, instead of convolving every pixel sequentially, the convolution kernel skips one pixel and performs the convolution as shown in Fig. 2. The amount of computation needed for this can be reduced by a factor of four, since the resolution of the features is reduced to half the width and height of the original feature.

The $4^{th} \sim 15^{th}$ layers are composed of six consecutive ResBlocks to efficiently train the low resolution features. Since the proposed model consists of deep convolutional layers, we have configured Resblock for most of the frameworks to avoid degradation problems that can occur during repetitive convolutional neural networks. The ResBlock is based on a conventional residual network with short connection [8]. This technique is well-known for its excellent ability to solve degradation problems, such as gradient vanishing or exploding by performing the feedforward method. As shown in Fig. 3, in our ResBlock, a short connection means

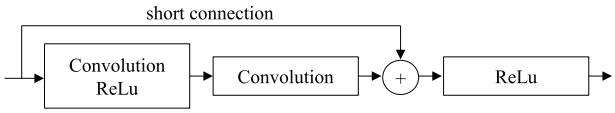

Fig. 3 Architecture of a ResBlock.

that the 64 features currently generated are combined with the 64 features created before the two convolutional layers.

To recover the original feature resolution, the 16^{th} layer uses a 3×3 deconvolution kernel with two stride, as shown in Fig. 2. First, each pixel in the feature is multiplied by the corresponding kernel weight and restored to the pixel position skipped by one pixel space. Next, the boundary value of the reconstructed pixels is the sum of the deconvolution results. The deconvolution method can perform up-sampling at the same time as learning, and it has the advantage of the fact that the loss caused by resampling of the features can be minimized [9].

The 64 features of the 16^{th} deconvolutional layer are combined with the 64 features of the 2^{nd} layer by long connection. The long connection tries to avoid degradation problems like other short connections, and it compensates for the loss caused by the two stride convolution and deconvolution. The final layer generates the noise canceled result image $DN(n_i)$ using the 3 × 3 convolution kernel.

4. Loss

We train our model with supervised learning using ground truth image data x_i. The means that we train our model using L2 loss between the ground truth x_i and the denoised image $DN(n_i)$. The model weights are updated using the loss value. By updating the weights using the loss value, the proposed DRNet generates an image very similar to the ground truth image. The supervised regression loss is defined by

$$loss = \frac{1}{2} \sum \|DN(n_i) - x_i\|^2 \quad (1)$$

5. Experiments

We use a tensorflow framework to train the proposed model. The simulated environment is running on a PC with Intel Core™ i7-6700K CPU 4GHz and Nvidia 1080Ti GPU. Also, we use Adaptive moment estimation (Adam) [10] with a learning rate of $lr = 0.001$, $\beta 1 = 0.9$, $\beta 2 = 0.999$, $\epsilon =$ 1e-08 and a mini-batch size of 128. We train 50 epochs for our model by using BSD400 image set containing 400 images. Also we use a BSD68 image set to test the model separately from the train image set and compare the result with other algorithms.

6. Result Analysis

We compare the simulation results with the existing noise reduction methods to demonstrate the superiority of our proposed method; BM3D [1], WNNM [2], EPLL [5] and

Table 2 Average PSNR comparison in BSD68 testset (unit: dB).

σ	BM3D [1]	WNNM [2]	EPLL [5]	DnCNN [7]	Proposed
15	30.07	31.37	31.11	31.73	32.01
25	28.57	28.83	28.45	29.23	29.63
50	25.62	25.87	24.80	26.23	26.60
Avg.	28.42	28.69	28.12	29.06	29.41

Table 3 Computing time comparison for images with different sizes (unit: s).

Method	Base Platform	256×256	481×321	512×512
BM3D [1]	CPU	0.7	1.9	3.1
EPLL [5]	CPU	31.16	76.89	130.75
DnCNN [7]	CPU	1.90	3.11	8.32
	GPU	0.02	0.05	0.09
Proposed	CPU	0.7	1.05	2.66
	GPU	0.01	0.02	0.03

DnCNN [7]. In order to make an the objective evaluation, we compare the peak signal-to-noise ratio (PSNR) value between the result image of the denoising algorithm and the ground truth image. In the simulation, we assign optimal Gaussian noise to ground truth images. The standard deviation of Gaussian noise was set by $\sigma = 15, 25, 50$.

Table 2 shows test results of PSNR comparison. According to these results, the proposed denoiser generates higher PSNR as much as about 0.99dB, 0.72dB, 1.29dB, 0.35dB than BM3D [1], WNNM [2], EPLL [5] and DnCNN [7] on average.

Table 3 shows a comparison of computing times. In this simulation, BM3D [1] and EPLL [5] run in the CPU environment. This result shows that the proposed denoiser generally eliminates noise in a shorter time than other methods.

Figure 4 shows simulation result of a real test image. The first image is a noisy image and the region of interest (ROI) is marked by a red box. From the result of BM3D [1], WNNM [2] and EPLL [5] in (b)–(d), the blurring effect is observed at the edge region. DnCNN [7] maintains sharper edges than other algorithms, but texture distortion has occurred and the animal legs on the back are not visible. From this result, we can see that our model can remove noise while preserving texture details.

7. Conclusion

In this paper, we have proposed a image denoiser using CNN with deconvolution and a modified residual network (DR-Net). We have analyzed that the diversity of initial features affects the performance of the whole algorithm. Therefore, the proposed algorithm extracts 128 features at the first layer based on 5×5 convolutional kernel. To reduce the computational complexity caused by the use of 128 features, our model trained downsized features by a two stride convolution. To keep identity and prevent gradient vanishing, we also used short and long connections at low and high resolutions, respectively. Experimental results show that the proposed algorithm produces faster and higher PSNR values than the other four algorithms while maintaining edge sharpness without blurring the texture.

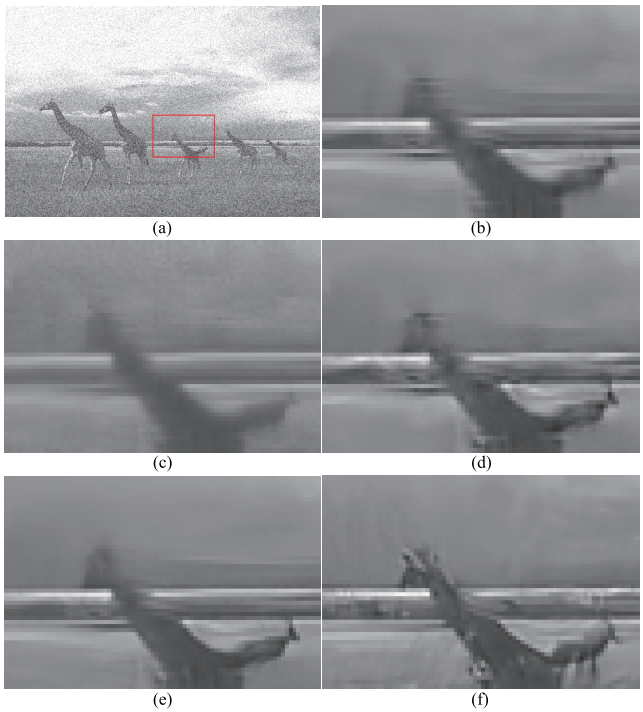

Fig. 4 The simulation result of "BSD68-test052", (a) noisy image $\sigma = 25$, (b) 31.63dB by BM3D, (c) 29.02dB by WNNM, (d) 30.17dB by EPLL, (e) 32.11dB by DnCNN, (f) 32.18dB by proposed algorithm.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2017R1D1A3B03034476) and research projects of "The Development of Security and Safety Systems based on Ubiquitous Technology for Shipping and Logistics".

References

[1] 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, 2007.
[2] S. Gu, L. Zhang, W. Zuo, and X. Feng, "Weighted nuclear norm minimization with application to image denoising," IEEE Int. Conf. Comput. Vis. Pattern Recognit., pp.2862–2869, 2014.
[3] Y.H. Guo and H.D. Cheng, "Image noise removal approach based on subpixel anisotropic diffusion," J. Electron. Imaging, vol.21, no.3, 033026, 2012.
[4] Y.-S. Zhang, F. Zhang, B.-Z. Li, and R. Tao, "Fractional domain varying-order differential denoising method," Opt. Eng., vol.53, no.10, 102102, 2014.
[5] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," IEEE Int. Conf. Comput. Vis., pp.479–486, 2011.
[6] J. Ryu and T. Nishimura, "Noise reduction in CMOS image sensor using cellular neural networks with a genetic algorithm," IEICE Trans. Inf. & Syst., vol.E93-D, no.2, pp.359–366, Feb. 2011.

[7] 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, 2017.

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

[9] M.D. Zeiler, G.W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," IEEE Int. Conf. Comput. Vis., pp.2018–2025, 2011.

[10] D.P. Kingma and J.L. Ba, "Adam: A method for stochastic optimization," Int. Conf. Learn. Representations, 2014.