Self-attentive Pyramid Network for Single Image De-raining

Taian Guo[1], Tao Dai[1,2](✉), Jiawei Li[1], and Shu-Tao Xia[1,2]

[1] Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, Guangdong,
People's Republic of China
gta17@mails.tsinghua.edu.cn, daitao.edu@gmail.com

[2] PCL Research Center of Networks and Communications, Peng Cheng Laboratory,
Shenzhen 518055, Guangdong, People's Republic of China

Abstract. Rain Streaks in a single image can severely damage the visual quality, and thus degrade the performance of current computer vision algorithms. To remove the rain streaks effectively, plenty of CNN-based methods have recently been developed, and obtained impressive performance. However, most existing CNN-based methods focus on network design, while rarely exploits spatial correlations of feature. In this paper, we propose a deep self-attentive pyramid network (SAPN) for more powerful feature expression for single image de-raining. Specifically, we propose a self-attentive pyramid module (SAM), which consists of convolutional layers enhanced by self-attention calculation units (SACUs) to capture the abstraction of image contents, and deconvolutional layers to upsample the feature maps and recover image details. Besides, we propose self-attention based skip connections to symmetrically link convolutional and deconvolutional layers to exploit spatial contextual information better. To model rain streaks with various scales and shapes, a multi-scale pooling (MSP) module is also introduced to efficiently leverage features from different scales. Extensive experiments on both synthetic and real-world datasets demonstrate the effectiveness of our proposed method in terms of both quantitative and visual quality.

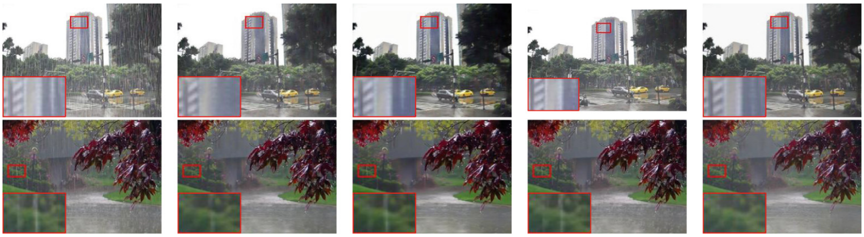
Keywords: Rain streak removal · Encoder-decoder network · Self-attention

1 Introduction

Images captured in rain weather are common in real life, thus resulting in images with rain streaks. Such rain streaks would not only affect the visual quality

This work is supported in part by the National Key Research and Development Program of China under Grant 2018YFB1800204, the National Natural Science Foundation of China under Grant 61771273, the R&D Program of Shenzhen under Grant JCYJ20180508152204044, and the research fund of PCL Future Regional Network Facilities for Large-scale Experiments and Applications (PCL2018KP001). We also gratefully acknowledge the support of NVIDIA Corporation with the donation of Titan X GPUs for this research.

T. Gedeon et al. (Eds.): ICONIP 2019, LNCS 11953, pp. 390–401, 2019.
https://doi.org/10.1007/978-3-030-36708-4_32



(a) Input (b) DDN (c) DID-MDN (d) PReNet (e) Ours

Fig. 1. Sample de-raining results on real-world rainy scenes with long heavy rain streaks. The details in the enlarged regions shows that our SAPN removes long heavy rain streaks in the input rainy images more cleanly, while keeps the sharp details of the background objects. The two rows demonstrate that our self-attentive network produces better de-raining results on image regions with long heavy rain streaks.

of images, but also degrade performance of existing computer vision systems, such as self-driving, video surveillance, and object detection. Therefore, it is of crucial importance to remove rain streaks while recovering image details. Image de-raining has received much attention in recent years, and can be generally divided into video-based [1–4] and single image based methods [5–10]. Most video based methods focus on utilizing the temporal correlations in successive frames, which provide extra temporal information of the rainy scene. In contrast, it is more challenging to perform single image de-raining due to the very limited information from a single image (Fig. 1).

In recent years, many single image de-raining methods [5–8,10] have been proposed. Most traditional image de-raining methods focus on exploiting powerful image prior of the rainy images, including sparse prior [7], low rank prior [11] and Gaussian mixture model (GMM) prior [6]. Among them, Luo et al. [5] proposed a dictionary learning based method, which sparsely approximates the patches of the rain layer and the de-rained layer by discriminative sparse codes with a learned dictionary. Li et al. [6] further introduced patch-based Gaussian mixture model (GMM) priors for both the background layer and the rain layer. Zhu et al. [7] introduced three types of priors, and proposed a joint optimization process to alternately remove rain-streak details. However, since such methods rely heavily on the handcrafted feature and the fixed priors, they are limited in practice due to the diversity of rain streaks (e.g., various shapes, scales and density levels).

Due to the powerful feature representation capability, convolutional neural networks have been widely used in image de-raining, and obtained remarkable performance. For example, Fu et al. [8] proposed a deep detail network to learn the high frequency details during the training process, since most rain streaks belong to high-frequency information. To consider various shapes and density of rain drops, Zhang et al. [10] proposed a densely connected network with learned rain streak density information to assist the rain streak removal process.

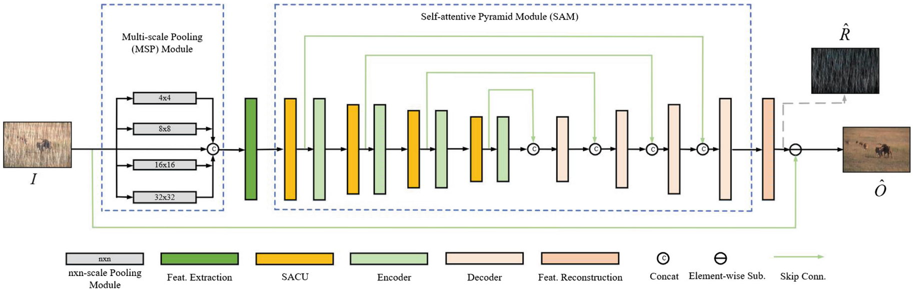


Fig. 2. Framework of our self-attentive pyramid network (SAPN)

Since spatial contextual information is important for rain streaks removal, some methods [12,13] have been developed. Specifically, Li et al. [12] proposed a multi-stage dilated CNN network to obtain a large receptive field size. Recently, Ren et al. [14] proposed a progressive recurrent network (PReNet) to better take advantage of the recursive computation and exploit the dependencies of deep features across stages.

Although significant progress has been achieved for single image de-raining, most of existing CNN-based methods focus on the network design, while rarely considering the inherent spatial correlations in feature maps. Meanwhile, self-attention [15] exploits the spatial correlations of features by using the attention scores to weight all features to obtain the salient features. To make full use of the spatial correlations of features, we propose a deep self-attentive pyramid network (SAPN) for single image de-raining, which mainly consists of self-attentive pyramid module (SAM) and multi-scale pooling (MSP) module. Specifically, to efficiently exploit the spatial contextual information, we propose the self-attention calculation units (SACUs) based encoding layers to enhance the encoding process, and SACUs based skip connections to enhance the symmetrical decoding process. With the assistance of SACUs, the encoding layers can better utilize the spatial correlations from the input features. Besides, our SACUs based skip connections can not only contribute to the propagation of gradient flows, but also pass the enhanced original feature signal from convolutional layers to symmetrical deconvolutional layers directly, which is helpful for recovering image details. Furthermore, since feature pyramid is helpful for multi-resolution feature representation, we apply multi-scale pooling in the shallow layers of our network. Extensive experiments on synthetic and real-world datasets demonstrate the superiority of our proposed method in terms of both quantitative and visual quality.

2 Self-attentive Pyramid Network (SAPN)

2.1 Network Architecture

As shown in Fig. 2, our SAPN consists of four main parts: multi-scale pooling module, shallow feature extraction, self-attentive pyramid module (SAM) and feature reconstruction.

Multi-scale Pooling Module. Given I as input rainy image and $\hat{R}$ as estimated rain streak image, then the output of SAPN is represented as follows:

$$\hat{O} = I - \hat{R}, \tag{1}$$

where $\hat{O}$ denotes the estimated de-rained image. To model rain streaks with various scales and shapes from the input image I, we firstly introduce a multi-scale pooling operation $H_{msp}(\cdot)$ to get the multi-scale feature F_{msp} from I:

$$\begin{aligned} F_{msp} &= H_{msp}(I) \\ &= [S_1^{32\times 32}(I), S_2^{16\times 16}(I), S_3^{8\times 8}(I), S_4^{4\times 4}(I), I], \end{aligned} \tag{2}$$

where $[\cdot, \cdot, ..., \cdot]$ denotes channel-wise concatenation and $S_i^{k\times k}(\cdot)$ represents the i-th $k \times k$-scale pooling operation which is defined as:

$$S_i^{k\times k}(I) = U^{k\times k}(ReLU(Conv^{1\times 1}(D^{k\times k}(I)))), \tag{3}$$

where $D^{k\times k}(\cdot)$ and $U^{k\times k}(\cdot)$ denote $k \times k$-scale downsampling and upsampling respectively. $Conv^{1\times 1}(\cdot)$ denotes a 1×1 convolutional layer.

Shallow Feature Extraction. After we get multi-scale feature concatenation F_{msp} from Eq. (3), the shallow feature representation F_{fr} can be obtained by

$$F_{fr} = H_{ex}(F_{msp}), \tag{4}$$

where $H_{ex}(\cdot)$ represents two 3×3 convolutional layers with 64 filters respectively, which are designed to extract the shallow feature representation F_{fr} from F_{msp}.

Self-attentive Pyramid Module (SAM). Given the shallow feature representation F_{fr} obtained from the above step, the self-attentive pyramid module (SAM), denoted as $H_{sam}(\cdot)$, adopts a pyramid encoder-decoder structure with Self-attention Calculation Units (SACU) embedded in it, and produce a rain streak layer feature representation F_{rs}:

$$F_{rs} = H_{sam}(F_{fr}). \tag{5}$$

The detailed description of SAM is given in Sect. 2.2.

Feature Reconstruction Part. After obtaining the rain streak layer feature representation F_{rs}, we can reconstruct the estimated rain streak $\hat{R}$ using the feature reconstruction part $H_{rc}(\cdot)$, which is actually a 3×3 convolutional layer:

$$\hat{R} = H_{rc}(F_{rs}) = H_{sapn}(I), \tag{6}$$

where $H_{sapn}(\cdot)$ represents the function of our proposed SAPN.

Loss Function. During the training process, our SAPN is optimized with loss function. To improve not only the pixel-wise reconstruction but the high-level semantic representation, we add perceptual loss to pixel-level L1 loss to get the combined loss L_C:

$$L_C = L_{L1} + \lambda L_P, \tag{7}$$

where λ denotes the trade-off coefficient between the two losses, and the L1 loss L_{L1} and the perceptual loss L_P are defined as:

$$L_{L1} = \frac{1}{CWH} \sum_{c=1}^{C} \sum_{w=1}^{W} \sum_{h=1}^{H} \|\hat{O}^{c,w,h} - O^{c,w,h}\|_1, \tag{8}$$

$$L_P = \frac{1}{CWH} \sum_{c=1}^{C} \sum_{w=1}^{W} \sum_{h=1}^{H} \|(V(\hat{O}))^{c,w,h} - (V(O))^{c,w,h}\|_2^2, \tag{9}$$

where C, W and H denote the channel, width and height dimension of the estimated de-rained image $\hat{O}$ and the ground truth clean image O. $V(\cdot)$ represents the front layers of a pretrained VGG model which is regarded as the high-level feature extractor. The loss function is optimized by Adam optimizer.

After a full glance at the framework of the proposed SAPN, we can conclude that the deep feature representation in our SAPN heavily relies on the self-attentive pyramid module (SAM), which will be shown in the next section.

2.2 Self-attentive Pyramid Module (SAM)

Our SAM is based on encoder-decoder networks [16], which are widely used in image-to-image tasks. However, most existing encoder-decoder based networks focus on network design, while rarely exploits spatial correlations of features and thus limits representation capability of the network. To exploit such correlations inherent in features, we propose a novel self-attentive pyramid module (SAM).

As shown in Fig. 2, the core component of SAPN is self-attentive Pyramid Module (SAM), which is further composed of four Self-attention Calculation Units (SACUs), four encoders and four decoders. The detailed description of SACU will be given in the next section.

Given the feature representation F_{fr} obtained from shallow feature extraction step $H_{ex}(\cdot)$, the original U-net [16] simply encodes the features iteratively and feeds the encoded features to symmetrical decoder. However, the single encoding layer, which consists of several convolutional layers, can not fully utilize the spatial correlations of the features, thus leading to poor ability of modeling the long-range dependency inherent in the features. Given this, we embed self-attention calculation unit (SACU) $H_{sa,i}(\cdot)$ in each encoder $H_{en,i}(\cdot)$ to model the long-range spatial correlations, and thus the encoded features are enhanced before passing through the decoding layer. i-th encoder $H_{en,i}(\cdot)$ is composed of a 3×3 convolutional layer with stride 2 and doubled channels from input, and two

3×3 layers with ReLU activation, which keeps input channels. We can formulate the self-attentive encoding part of the SAM component $H_{sam}(\cdot)$ as:

$$\begin{aligned} F_{sa,i} &= H_{sa,i-1}(F_{en,i-1}), \\ F_{en,i} &= H_{en,i}(F_{sa,i}),\ i=1,2,3,4, \end{aligned} \tag{10}$$

where $F_{sa,i}$ denotes the output of i-th SACU and $F_{en,i}$ denotes the output of i-th encoder. $F_{en,0}$ denotes F_{fr} for convenience. With the help of self-attention information obtained from the SACU, the encoding process can be enhanced to get more spatial correlation into consideration.

After the self-attentive encoding part, we obtain $F_{sa,i}$ plus the final output (i.e. $F_{en,4}$) of encoders as the input of following decoding part. Unlike the pyramid network and U-net, which directly utilize the symmetrical encoded features $F_{en,4-i}$ from skip connection as the extra information:

$$F_{de,i} = H_{de,i}([F_{de,i-1}, F_{en,4-i}]), \tag{11}$$

we adopt the extra self-attention information besides the original encoded features $F_{en,4-i}$, which is integrated in features $F_{sa,4-(i-1)}$, to decoder $H_{de,i}(\cdot)$ to get output features $F_{de,i}$. Similar with the encoder design, i-th decoder $H_{de,i}(\cdot)$ starts with a 3×3 deconvolutional layer with stride 2 and keeps the channels, followed by two consecutive 3×3 layers with ReLU activation which halves the channels in the former layer. Specifically, we utilize the obtained self-attention $F_{sa,i}$ to enhance the decoding process, which can be formulated as:

$$F_{de,i} = H_{de,i}([F_{de,i-1}, F_{sa,4-(i-1)}]), \tag{12}$$

where $F_{de,4}$, the output features of the last decoder, is also the final output of SAM and input of feature reconstruction layer, which is also denoted as F_{rs}. $F_{de,0}$, or $F_{en,4}$, is the output features of the last encoding layer and also the input of the first decoding layer. Through the skip connection which delivers the output self-attention $F_{sa,4-(i-1)}$ of $(4-i)$-th SACU (i.e. $H_{sa,4-(i-1)}(\cdot)$), the i-th decoder $H_{de,i}(\cdot)$ can get not only the symmetrical features but their self-attention information directly since the skip connection structure in SACU. With the help of the extra self-attention information of features, the decoding process can be further enhanced with the spatial correlation provided by the self-attention information, which makes the representation of long-range dependency possible. The experiments in Sect. 3 demonstrate the performance gain of the utilization of the extra self-attention information. We will give a further explanation to the self-attention calculation unit (SACU) in the next section.

2.3 Self-attention Calculation Unit (SACU)

Self-attention focuses on the attention of feature maps towards themselves, which has been widely researched by previous works [15,17]. The information provided by self-attention properly handles the problem that long-range feature dependency can not be efficiently convolved by the convolutional layers.

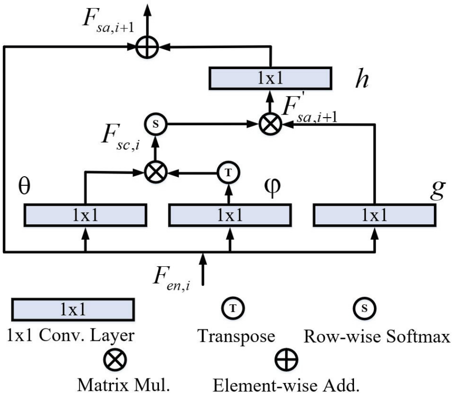


Fig. 3. Self-attention Calculation Unit (SACU)

As shown in Fig. 3, given the output features $F_{en,i}$ of i-th encoder $H_{en,i}(\cdot)$, we firstly obtain three embedding representation $\theta(F_{en,i})$, $\phi(F_{en,i})$, $g(F_{en,i})$ from three different 1×1 convolutional layers $\theta(\cdot)$, $\phi(\cdot)$ and $g(\cdot)$. Then the self-correlation $F_{sc,i}$ of feature map $F_{en,i}$ can be obtained via

$$F_{sc,i} = \theta(F_{en,i})\phi(F_{en,i})^{\mathrm{T}}. \tag{13}$$

Then, we can get the self-weights $F^{'}_{sc,i}$ by softmaxing each row of $F_{sc,i}$, and use it to weight the embedding representation $\theta(F_{en,i})$ by:

$$F^{'}_{sa,i+1} = F^{'}_{sc,i}g(F_{en,i}). \tag{14}$$

After that, the self-attention map $F^{''}_{sa,i+1}$ is obtained via a further 1×1 convolutional layer $h(\cdot)$.

Furthermore, to boost the gradient transmission and avoid the gradient vanishing problem, we add skip connection from the input $F_{en,i}$ to the calculated self-attention map $F^{''}_{sa,i+1}$. To better calibrate the influence between them, unlike the original non-local implementation [17], which regards the balance between the two terms as a hyper-parameter, we bring in a learnable parameter α as a trade-off weight. The final output $F_{sa,i+1}$ of SACU can be formulated as:

$$F_{sa,i+1} = F_{en,i} + \alpha F^{''}_{sa,i+1}. \tag{15}$$

With the learnable parameter α, the weighting of the self-attention information becomes more flexible and thus leads to better utilization of self-attention.

3 Experiments

To validate the advantage of our method, we conduct tremendous experiments on various synthetic datasets and natural rainy images. Since the ground truth images are available in synthetic datasets, PSNR and SSIM are adopted as the evaluation criterion of the de-raining results. We calculate PSNR/SSIM in luminance channel of YCbCr space. Additionally, we compare our proposed SAPN with state-of-the-art de-raining methods, including Deep Detailed Network (DDN) [8], Joing Rain Detection and Removal (JORDER) [9], Density-aware Single Image De-raining using a Multi-stream Dense Network (DID-MDN) [10] and Progressive Recurrent Network (PReNet) [14].

3.1 Datasets

Synthetic Datasets. To make a comparison with previous state-of-the-art de-raining approaches, we adopt three public benchmark synthetic datasets to train and evaluate our SAPN, including DDN-Dataset [8], DID-MDN-Dataset [10] and Rain100H [9]. Specifically, DDN-Dataset contains 14,000 rainy-clean image pairs which is synthesized by 1000 clean images. We randomly select 9100 image pairs as training dataset and use the left 4900 image pairs as the testing dataset. DID-MDN-Dataset is composed of 12000 training rainy-clean image pairs and 1201 testing image pairs. Rain100H contains 100 testing images and there are 1800 training image pairs in the corresponding training dataset (i.e. RainTrainH).

Real-World Images. To validate the effectiveness of the proposed network in real world rainy scenes, we randomly select some images from the previous de-raining works [8,9,18,19] and the internet.

3.2 Training Details

For each of the three datasets, we train our SAPN on a 1080 ti GPU on the training dataset, and evaluate the model on corresponding testing dataset. We train our model for 300, 350 and 50 epochs for DID-MDN-Train, DDN-Train, and RainTrainH respectively. The initial learning rate is set to $2 \cdot 10^{-4}$ and decreased linearly at the end of every epoch. To avoid the problem of over-fitting, we use a weight decay of 10^{-5} and Adam optimizer with betas 0.5 and 0.999. The model is trained on the Pytorch framework.

Table 1. Quantitative results of average PSNR (dB)/SSIM compared with state-of-the-art de-raining works. The two best-performing methods are marked in **bold** and underlined respectively.

Dataset	Matric	Input	DDN [8] (CVPR'17)	JORDER [9] (CVPR'17)	DID-MDN [10] (CVPR'18)	Our SAPN
DID-MDN-Dataset	PSNR (dB)	23.63	30.08	26.80	29.36	**30.86**
	SSIM	.7313	.8788	.8361	.9002	**.9230**
DDN-Dataset	PSNR (dB)	23.74	30.00	26.47	28.00	**30.26**
	SSIM	.7499	.8932	.8276	.8776	**.9110**
Rain100H	PSNR (dB)	13.56	22.26	26.10	26.35	**27.06**
	SSIM	.3800	.6928	.7971	.8287	**.8474**

3.3 Results on Synthetic Datasets

The details of evaluation results on synthetic datasets are shown in Table 1. Note that the pretrained model of PResNet [14] is trained from other datasets, thus we did not report the quantitative results for fair comparison. Results show that our method outperforms other state-of-the-arts consistently. This is mainly because our SAPN utilizes both multi-scale information and spatial correlations of features, which enhances the feature representation capability of the network. In contrast, DDN [8] learns the mapping from high-frequency details of rainy

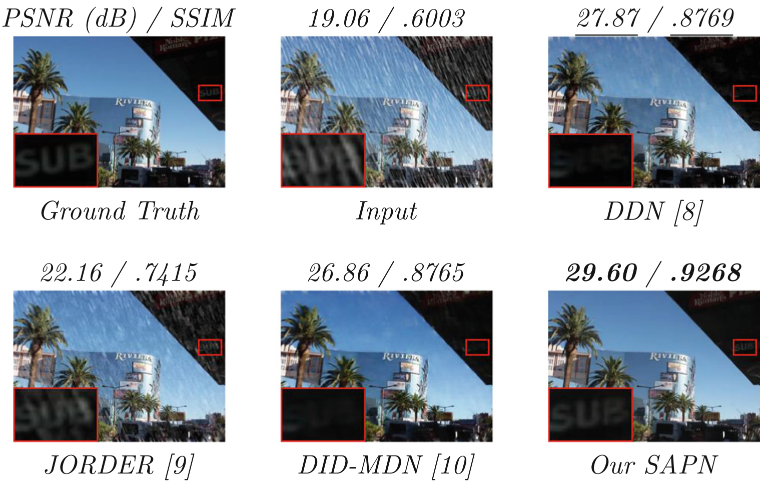


Fig. 4. De-raining results on sample image from DDN-Testset.

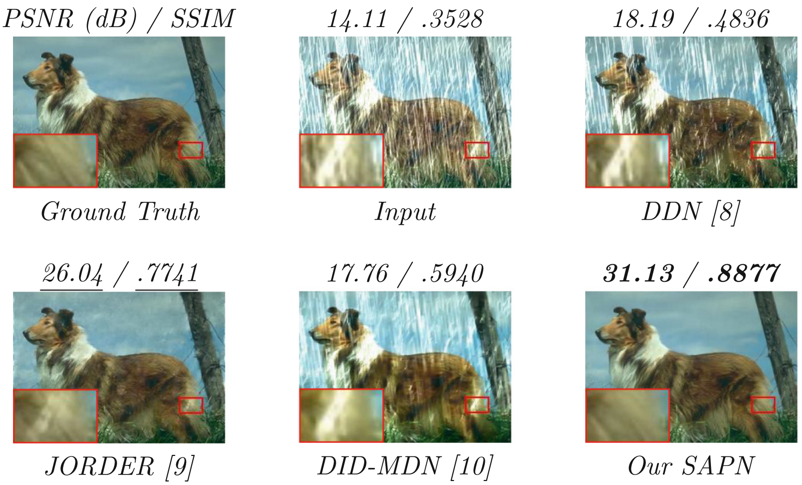
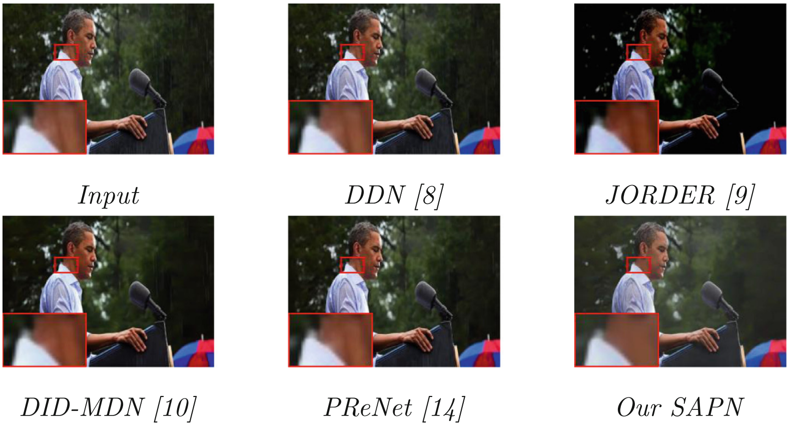


Fig. 5. De-raining results on sample image from Rain100H.



Input *DDN [8]* *JORDER [9]*

DID-MDN [10] *PReNet [14]* *Our SAPN*

Fig. 6. De-raining results on sample real-world image.

images to clean ones with ResNet [20], while DID-MDN [10] utilize multi-scale features and multi-stream DenseNet [18] architecture, both of which do not take spatial correlations inherent in features into consideration.

Besides the quantitative evaluation on synthetic datasets, we also randomly select several images from the testing datasets to validate the visual effect. As shown in Figs. 4 and 5, our method obtains better visual results. For example, the alphabets in enlarged region in Fig. 4 are clearly recovered by our SAPN, while other methods fail to remove long heavy rain streaks or bring in unpleasant artifacts. Another sample in Fig. 5 also show that our SAPN keeps the background scenes better and removes the rain streaks more cleanly, especially when the image has some long rain streaks or other objects with long shapes, since the adoption of self-attention mechanism enhances the capability of the network to capture long-range dependency and non-local similarity.

3.4 Qualitative Evaluation on Real-World Images

To verify the performance gain of SAPN over previous methods on rainy scenes in real world, we also test our SAPN and other methods on real-world images. The de-raining results on a randomly selected real world image sample is shown in Fig. 6. Noticeably, our method achieves extremely better results when the rain streaks in rainy image are longer than average, just because we adopt self-attention mechanism in our network design, which can better leverage non-local similarity of input rainy image and attain long-range dependency more effectively and more efficiently. This specialty of our SAPN helps locate rainy areas in input rainy images, leading to better final de-raining results. It is clearly shown in Fig. 6 that our SAPN produces preferable results compared with other methods, which tend to either under de-rain or over de-rain the natural rainy images. Specifically, all other four methods fail to remove all long rain streaks, while JORDER even brings in severe artifacts. In contrast, our method not only removes more rain streaks, but preserves background details better.

3.5 Ablation Study

To verify benefits of each individual component, including multi-scale pooling (MSP) module and SACUs, we train some variants of our SAPN on RainTrainH and evaluate trained models on Rain100H. The results are shown in Table 2.

Table 2. Ablation study of our proposed SAPN on SACUs and multi-scale pooling module on Rain100H.

Methods	U_a	U_b	U_c	U_d
SACUs?		✓		✓
MSP?			✓	✓
PSNR (dB)	26.78	26.95	26.89	27.06

We can conclude that the adoption of SACUs effectively promotes the de-raining results of the basic pyramid encoder-decoder network (U_a), while MSP also improves the performance of the network effectively. The combination of SACUs and MSP leads to our final SAPN architecture (U_d).

4 Conclusion

In this paper, we propose a pyramid encoder-decoder network with self-attention calculation units for single image de-raining. Compared with previous methods which does not exploit spatial correlations of features, our method explicitly learns the self-correlation inherent in output features of each encoder layer, making the encoding and symmetrical decoding process more self-attentive and better resolve the long-range dependency problem in images, leading to better eventual de-raining results, especially in long rain streaks conditions. In order to further improve the de-raining results, we add a multi-scale pooling module before feature extraction, which leads to even higher quantitative performance and much better visual experience. Tremendous experiments on various datasets validate that our network outperforms the state-of-the-art methods.

References

1. Bossu, J., Hautière, N., Tarel, J.-P.: Rain or snow detection in image sequences through use of a histogram of orientation of streaks. IJCV **93**, 348–367 (2011)
2. Ren, W., Tian, J., Han, Z., Chan, A., Tang, Y.: Video desnowing and deraining based on matrix decomposition. In: CVPR (2017)

3. Wei, W., Yi, L., Xie, Q., Zhao, Q., Meng, D., Xu, Z.: Should we encode rain streaks in video as deterministic or stochastic. In: CVPR (2017)
4. Garg, K., Nayar, S.K.: Detection and removal of rain from videos. In: CVPR (2004)
5. Luo, Y., Xu, Y., Ji, H.: Removing rain from a single image via discriminative sparse coding. In: ICCV (2015)
6. Li, Y., Tan, R.T., Guo, X., Lu, J., Brown, M.S.: Rain streak removal using layer priors. In: CVPR (2016)
7. Zhu, L., Fu, C.-W., Lischinski, D., Heng, P.-A.: Joint bilayer optimization for single-image rain streak removal. In: ICCV (2017)
8. Fu, X., Huang, J., Zeng, D., Huang, Y., Ding, X., Paisley, J.: Removing rain from single images via a deep detail network. In: CVPR (2017)
9. Yang, W., Tan, R.T., Feng, J., Liu, J., Guo, Z., Yan, S.: Deep joint rain detection and removal from a single image. In: CVPR (2017)
10. Zhang, H., Patel, V.M.: Density-aware single image de-raining using a multi-stream dense network. In: CVPR (2018)
11. Chang, Y., Yan, L., Zhong, S.: Transformed low-rank model for line pattern noise removal. In: ICCV (2017)
12. Li, X., Wu, J., Lin, Z., Liu, H., Zha, H.: Recurrent squeeze-and-excitation context aggregation net for single image deraining. In: ECCV (2018)
13. Li, G., He, X., Zhang, W., Chang, H., Dong, L., Lin, L.: Non-locally enhanced encoder-decoder network for single image de-raining. In: MM (2018)
14. Ren, D., Zuo, W., Hu, Q., Zhu, P., Meng, D.: Progressive image deraining networks: a better and simpler baseline. In: CVPR (2019)
15. Vaswani, A., et al.: Attention is all you need. In: NIPS (2017)
16. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
17. Wang, X. Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: CVPR (2018)
18. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR (2017)
19. Zhang, H., Sindagi, V., Patel, V.M.: Image de-raining using a conditional generative adversarial network. arXiv (2017)
20. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)