Pyramid fully residual network for single image de-raining

Guangle Yao\textsuperscript{a}, Cong Wang\textsuperscript{b,\textdegree}, Yutong Wu\textsuperscript{c}, Yang Wang\textsuperscript{d}

\textsuperscript{a}Chengdu University of Technology, Chengdu, China
\textsuperscript{b}The Hong Kong Polytechnic University, Hong Kong, China
\textsuperscript{c}Dalian University of Technology, Dalian, China
\textsuperscript{d}Nanjing University of Aeronautics and Astronautics, Nanjing, China

\textbf{Abstract}

Rain removal from a single image is a challenging and significant task of image pre-processing. In this paper, we learn the multi-scale streaks from rainy images using feature pyramid, and to improve the effectiveness of the learning, we focus on the feature propagation re-usage and propagation in the extremely deep de-raining network, respectively, called Pyramid Fully Residual Unit and Network (PFR-Unit and PFR-Net). The PFR-Unit employs fully residual learning in each level of feature pyramid and the PFR-Net connects PFR-Units by a compact dense architecture. The fully residual learning encourages the feature re-usage in PFR-Unit by performing identity mapping for all available shortcuts. The compact dense connection strengthens the feature propagation between the PFR-Units and ensures the unicity of the learning space for the PFR-Units. Along with negative SSIM loss, the PFR-Net presents a good performance in single image de-raining. Comprehensive experimental results show that the PFR-Net outperforms the state-of-the-art single de-raining methods with a big margin on Rain100H, Rain100L and Rain1200 datasets.

1. Introduction

As a kind of common weather, rain often degrades the visibility and decreases the performance of outdoor computer vision systems, ranging from video surveillance, robotic to self-driving vehicle. However, the existing models of classification, detection and segmentation in these systems are trained on the image or video samples without rain. Thus, it is necessary to design effective methods to remove the rain streaks from the rainy images. Since 2012, Convolutional Neural Network (CNN) \cite{1}, which learns the trainable filters, achieved great success and has been widely employed in the tasks of vision and multimedia, significantly improving the performance of image classification \cite{2}, object detection \cite{3}, scene classification \cite{4}, de-hazing \cite{5,6} and single image de-raining.

In this paper, we incorporate an end-to-end deep CNN, named PFR-Net, to address the problem of single image de-raining by modeling a pixel-wise image regression process, which is formalized as:

\[ B = O - R, \]  

where \( O \) and \( B \) denote rainy image, rain streaks image and the de-rained image, respectively. In PFR-Net, we learn the multi-scale rain streaks using feature pyramid. And to improve the performance of this deep network and make it easy to train, we boost the feature re-usage and feature propagation, respectively, by the introduced fully residual learning and compact dense connection. Fig. 1 shows an example of the learned rain streaks image and de-rained image by our PFR-Net.

\textbf{Fully Residual Learning.} Residual Learning \cite{7} re-uses the features by performing identity mapping to improve the performance from considerably increased depth. Without consideration of the activation operations, a residual block with 2 convolution operations in Fig. 2 (a) is defined as:

\[ x_2 = x_0 + \text{Conv}_2(\text{Conv}_1(x_0)), \]  

where \( \text{Conv} \) is the convolution operation. The introduced fully residual learning for this block is illustrated in Fig. 2 (b), which performs identity mapping for three available shortcuts and can be formalized as:

\[ x_2 = x_0 + \text{Conv}_1(x_0) + \text{Conv}_2(x_0 + \text{Conv}_1(x_0)), \]  

In Fig. 2 (c), we generalize this fully residual learning for a block with \( n \) convolution operations and activation function \( Q \) as a recursive function:
\[ x_t = Q(x_0 + \cdots + x_{t-1}) + \text{Comp}_t(x_0 + \cdots + x_{t-1}), \] (4)

Compared with residual learning, the fully residual learning performs identity mapping for all available shortcuts to enhance the feature re-usage.

**Compact Dense Connection.** Dense connection \([8]\) has been proven an efficient architecture by strengthening feature propagation. Given the composite function \(H_l\) with \(l \times C\) channels, the \(l^{th}\) \(H\) receives the feature maps of all preceding \(H\) as input:

\[ X_l = H_l(\text{Concat} x_0; x_1; \cdots; x_l), \] (5)

where \(\text{Concat} x_0; x_1; \cdots; x_l\) refers to the concatenation of the feature maps produced by \(H_0, H_1, \cdots, H_l\). In this paper, we introduce a compact dense architecture to connect the CNN-based blocks, such as the designed PFR-Unit, as follows:

\[ X_l = \text{Comp}_l(\text{Concat} x_0; x_1; \cdots; x_{l-1}; H_l(x_l)), \] (6)

where \(H_S\) is the CNN-based block with \(C\) channels, \(\text{Concat} x_0; x_1; \cdots; x_{l-1}\) refers to the concatenation of the output of all preceding \(H_S\) blocks, and \(\text{Comp}_l\) is the compact operation to convert the channel from \(l \times C\) to \(C\) for the \(l^{th}\) \(H_S\) block. We employ a trainable \(1 \times 1\) convolution as the compact operation. The dense connection boosts the information flow between \(H_S\) blocks. And the compact operation reduces the parameters of the \(H_S\) block and ensures the unicity of the learning space for \(H_S\) block.

Fig. 3 presents our de-raining method. The designed PFR-Unit employs fully residual block in each level of feature pyramid. And the proposed PFR-Net connects the PFR-Units with compact dense architecture. By dense connection, this architecture strengthens the feature propagation between the PFR-Units, and by compacting channels, it makes the fully residual learning possible to re-use the features in PFR-Unit. Additionally, this architecture ensures the single type of learning space for the PFR-Unit with limited parameters.

In summary, this paper makes the following contributions:

- We introduce a fully residual connection and design a powerful de-raining unit. The de-raining unit fully re-uses the features in a pyramid to learn multi-scale rain streaks.
- We introduce a compact dense architecture to connect the CNN-based blocks, such as de-raining unit, which strengthens the feature propagation between the connected blocks and ensures the unicity of learning space for the blocks.
- We propose an effective deep de-raining network by encouraging feature re-use and feature propagation, which achieves superior performance over state-of-the-art single image de-raining methods.

2. Related works

In the past few decades, a large number of de-raining methods have been proposed. The early research \([9–11]\) focused on rain removal from videos by exploiting temporal correlation between video frames. Tripathi et al. \([12]\) provided an overview of a series of video-based de-raining methods. In recent years, the research \([13–26]\) on rain removal from single image gained more attention. Compared with video-based de-raining, single image de-raining is more challenging due to the lack of temporal correlation information. Same with other computer vision tasks, the single image de-raining in the literature can be grouped into: (1) model-driven methods and (2) data-driven methods.

**Model-driven methods:** The model-driven methods explore prior knowledge to address the problem of de-raining. Kang et al. \([13]\) assumed that rain streaks are high frequency structure and separated the rain streaks by utilizing sparse coding from HOG features in high frequency layer. Luo et al. \([14]\) proposed a discriminative sparse coding framework based on image patches and separated rain streaks from rain-free background images. Ding
et al. [27] designed guided filter [28] to obtain coarse rain-free image and recovered the refined result by a further minimization operation. Li et al. [15] proposed to exploit GMMs which act as patch-based priors to separate the rain streaks layer from background images.

Data-driven methods: The data-driven methods started from 2017 which adopt the deep neural network architectures to learn information from the rainy-background image pairs to solve the de-raining problem. Fu et al. [17] decomposed rainy images into low- and high-frequency parts by guided filter and mapped high-frequency parts to rain streaks by a residual network to remove the rain streaks. Yang et al. [18] designed a recurrent contextual convolutional neural network to jointly detect and remove rain streaks. Li et al. [29] proposed a non-local enhanced encoder-decoder network that maps rainy image to clean image via learning the residual. Li et al. [19] put forth a recurrent squeeze-and-excitation [30] context aggregation net for single image de-raining. Zhang et al. [20] presented a density-aware guided multi-stream connected network to jointly estimate rain density and clean images. Hu et al. [31] designed an end-to-end deep neural network to learn depth-attentional features via a depth-guided attention mechanism and regress a residual map to produce the rain-free image output. Li et al. [32] designed a 2-stage network: a physics-based backbone to estimate the rain streaks and a depth-guided GAN to recover the background details. Guo et al. [33] proposed a multi-scale neural architecture search method for image restoration which remarkably alleviate the difficulty in architecture design.

3. Proposed method

As illustrated in Fig. 3, the proposed de-raining PFR-Net is an end-to-end convolutional neural network which is composed of the PFR-Units and learns the complex pixel-wise mapping from the rainy-background image to the de-rained image. In this section, we provide a description of the proposed de-raining unit and network.

3.1. PFR Network

Due to the simpler structure, the rain streaks is easier to learn than background. The PFR-Net learns the rain streaks image $\hat{R}$ from
the rainy image $O$ and obtains the de-rained image $\hat{B}$ via a subtraction operation as described in Eq. 1. Firstly, a $3 \times 3$ convolution operation $Conv_{3 \times 3}$ is imposed on the rainy image $O$ to encode the image space with 3 channels to the feature space with $C$ channels:

$$F_0 = Conv_{3 \times 3}(O).$$

(7)

Then the introduced compact dense architecture with PFR-Unit works in the feature space to learn the feature of rain streaks:

$$F_1 = Comp_{l}(Concat[F_0, F_1, \ldots, F_{l-1}, PFR(F_{l-1})]),$$

(8)

where $F_{l-1}$ is the input of the $l$th PFR-Unit, $l = 1, 2, \ldots, L$. By dense connection, this architecture strengthens the feature propagation between PFR-Units to improve the performance of PFR-Net and benefits the gradients pass throughout the PFR-Net to make the network easy to train. As the compact operation, the $1 \times 1$ convolution compacts the input channel of the $l$th PFR-Unit from $l \times C$ to $C$ and makes the fully residual learning in PFR-Unit possible. Additionally, this compact operation reduces the parameters of PFR-Units and decreases the diversity of the learning space for the PFR-Units. The final rain streaks feature map $F_L$ is produced by this compact dense architecture with $L$ PFR-Units.

Beyond the learning by compact dense architecture, there is a $3 \times 3$ convolution $Conv_{3 \times 3}$ which decodes the feature map $F_L$ with $C$ channels to the 3 channels rain streaks image $\hat{R}$:

$$\hat{R} = Conv_{3 \times 3}(F_L).$$

(9)

Finally, the final de-rained image $\hat{B}$ is obtained by removing the rain streaks $\hat{R}$ from rainy image $O$:

$$\hat{B} = O - \hat{R}.$$

(10)

3.2. PFR Unit

Recently, numerous spatial feature pyramid in neural network architectures are proposed to resolve the multi-scale challenge for kinds of computer vision tasks. One important strategy creates feature pyramid with the low-resolution features and high-resolution features of neural network via top-down pathway. Lin et al. [34] proposed a feature pyramid network for object detection and Chen et al. [35] came up with a cascaded pyramid network for multi-person pose estimation. Another commonly used strategy generates feature pyramid by pooling operation on the same input features. Zhao et al. [36] developed a multi-scale spatial pooling scheme and combined it with a multi-channel attention selection for cross-view image translation. Tang et al. [37] designed a pyramid pooling module in the scene parsing network. Our PFR-De-raining Unit in Fig. 4 naturally leverage the conventional feature pyramid to learn the rain steaks at vastly different scales by pooling operation. Different with the pyramid pooling schemes in [36,37] which generate multi-scale features and fuse these features directly, the PFR-Unit performs much more learning from each scale feature before fusion by the proposed fully residual. On the other hand, there is only one pyramid pooling scheme imposed on the feature extracted by deep network in [36,37], but a series of pyramid pooling schemes, PFR-Units in our method, connected by the proposed compact dense architecture to form deep network for continuous multi-scale learning.

Firstly, pooling operations with different kernels and strides are used to obtain multi-scale features:

![Fig. 4. PFR-Unit.](image-url)
where $F$ denotes the input of PFR-Unit. $Pooling_k$ denotes the Max pooling operation with the kernel $2^{k-1} \times 2^{k-1}$ and stride $2^{k-1} \times 2^{k-1}$. $S_k$ denotes the scaled feature map in the $k^{th}$ level of the feature pyramid. In each level, the fully residual learning with convolution operations and activation functions is imposed on the scaled feature map $S_k$:

$$Z_k = \text{LeakyReLU}(S_k + S_{k-1} + \cdots + S_{N} + \text{Con}v_0(S_{k-1} + S_{k-2} + \cdots + S_N))$$

(12)

where $\text{Con}v_N$ is the $2^{N} \times 3 \times 3$ convolution with $C$ channels in the fully residual block. $\text{LeakyReLU}$ is the non-linear activation function. $Z_k$ is the output of fully residual block in the $k^{th}$ level. The final output feature map of the PFR-Unit is the combination of the feature maps $Z_k$ by a concatenation and a $1 \times 1$ convolution operation:

$$Z = \text{Con}v_{1,1}(\text{Concat}[U_{p0}(Z_0), \cdots, U_{pN}(Z_N)])$$

(13)

where $Z$ is the output of PFR-Unit. $U_{pN}$ is a $2^{N-1} \times 1 \times 1$ upsampling operation to produce the features from different levels with same spatial sizes. $\text{Concat}$ refers to the concatenation of the upsampled features. And the $\text{Con}v_{1,1}$ is the $1 \times 1$ convolution operation which merges the multi-scale steak information by converting the channel from $K \times C$ to $C$ and also reduces the aliasing effect of upsampling.

3.3. Loss function

MSE (Mean Square Errors) measures pixel-wise errors without considering local image characters, e.g., edge, local contrast and luminance, which are sensitive to human visual system. In contrast, SSIM (structural similarity index) [38] is calculated based on these local image characteristics, which are also the characteristics of rain steaks. Thus, SSIM is much more appropriate to guide to train the de-raining network. A bigger SSIM value indicates that the de-rained image is closer to the ground truth. In training phase, with the deep neural network converging, the loss should became smaller. Therefore, negative SSIM loss as in Eq. 14 is empirically used for de-raining task and has been proved a better performance [21] in terms of both SSIM and PSNR (Peak Signal to Noise Ratio) [39].

In this paper, we employ the negative SSIM loss to train the PFR-Net:

$$\mathcal{L} = -\text{SSIM} (\hat{B}, B)$$

(14)

where $\hat{B}$ and $B$ are the de-rained image and the corresponding ground-truth background.

3.4. Implementation Details

We connect 20 PFR-Units with the compact dense architecture. And for the feature pyramid of the PFR-Unit, we perform the fully residual learning using $4 \times 3 \times 3$ convolutions in each level and set the channel of these convolutions to 32. We randomly select 100 $\times$ 100 patch pairs from training image datasets as input of the PFR-Net and set the number of level to 4 for the feature pyramid. For all the $3 \times 3$ and $1 \times 1$ convolutions in PFR-Net are followed by a non-linear activation LeakyReLU with $\alpha = 0.2$. Our PFR-Net is implemented using PyTorch and trained with Adam algorithm [40] using 4 NVIDIA TITAN XP GPUs. We initialize the learning rate to 0.0005, and divide it by 10 at 1200 epochs and 1500 epochs, and stop the training at 2000 epochs.

4. Experiments Results

In this section, we conduct extensive experiments on three widely used datasets to compare our methods with several state-of-the-art methods: DSC (ICCV2015) [14], LP (CVPR2016) [15], DDN (CVPR2017) [17], RESCAN (ECCV2018) [19], NLEDN (ACM MM2019) [29], REHEN (ACM MM19) [41], PreNet (CVPR2019) [21], SpaNet (CVPR2019) [22].

4.1. Datasets and Evaluation Metrics

**Synthetic Datasets:** Rain100H [18], Rain100L [18] are Rain1200 [20] are widely used synthetic datasets. They have various rain streaks, including different sizes, shapes and directions. Both of Rain100H and Rain100L include 1800 pairs of images for training and 200 pairs for testing. Rain1200 has 12000 and 1200 pairs, respectively, for training and testing. All testing images are assured to have different background images with training images. We also select Rain100H for the further analysis.

**Real-world Datasets:** [42,18] provided some datasets including real-world rainy images and we also download a number of rainy images from Google. We conduct the experiment to demonstrate the effectiveness of the proposed PFR-Net on these real-world images.

**Evaluation Metrics:** PSNR and SSIM are widely used to measure the quality of the restored image, which are used as the evaluation metrics in the experiments on synthesized data. Since there is no ground-truth reference for real-world images, the performance of the proposed and compared methods on the real-world dataset is evaluated visually.

4.2. Results on Synthetic Datasets

We provide the results of our method and state-of-the-art methods on three synthetic datasets in Table 1, which shows that our method outperforms the compared methods with a big margin. Especially, compared with two methods in CVPR’19, our approach improves the PSNR, respectively, by 2.74 dB and 4.09 dB on Rain100H, the most challenging dataset. And the SSIM is improved by 4% and 3%, respectively. Furthermore, we present several visual examples in Fig. 5 and 6. Fig. 5 presents the comparison of our method with two model-driven methods. It is obvious that the results of model-driven methods are unacceptable, while our restored image is clean and clear. Fig. 6 present the comparison of our method with state-of-the-art data-driven methods. It can be observed that our method obtains the best de-raining performance, while the compared state-of-the-art methods maintain a number of artifacts or rain streaks.

4.3. Results on Real-world Datasets

We further evaluate the performance of our method on real-world datasets and show the results in Fig. 7–9. Same with the evaluation on synthetic dataset, the results of model-driven methods on real-world dataset are still unacceptable and data-driven methods present better results. In Fig. 8 and 9, we also enlarge some selected region from the de-rained images produced by data-driven methods. Among these methods, no matter from the overall perspective or these selected regions, our method achieves the best performance, which removes nearly all the rain streaks with different scales and shapes and obtains the clearest and cleanest de-rained images with more details of the texture.
Table 1
Quantitative experiments evaluated on three synthetic datasets. The best results are highlighted in boldface.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>DSC ICC’15</th>
<th>LP CVPR’16</th>
<th>DDN CVPR’17</th>
<th>RESCAN ECCV’18</th>
<th>NLEDN ACM MM’18</th>
<th>REHEN ACM MM’19</th>
<th>PreNet CVPR’19</th>
<th>SpaNet CVPR’19</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rain100H</td>
<td>15.66 0.42</td>
<td>14.26 0.54</td>
<td>22.26 0.69</td>
<td>25.92 0.84</td>
<td>28.42 0.88</td>
<td>27.52 0.86</td>
<td>26.54 0.90</td>
<td>30.40 0.93</td>
<td></td>
</tr>
<tr>
<td>Rain100L</td>
<td>24.16 0.87</td>
<td>29.11 0.88</td>
<td>34.85 0.95</td>
<td>36.12 0.97</td>
<td>38.84 0.98</td>
<td>37.91 0.98</td>
<td>36.20 0.98</td>
<td>39.07 0.99</td>
<td></td>
</tr>
<tr>
<td>Rain1200</td>
<td>21.44 0.79</td>
<td>22.46 0.80</td>
<td>30.95 0.86</td>
<td>32.35 0.89</td>
<td>32.98 0.92</td>
<td>32.51 0.91</td>
<td>30.01 0.92</td>
<td>33.17 0.93</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 5. Comparison of our method with model-driven methods on synthetic dataset.

Fig. 6. The results from synthetic datasets. From left to right: (a) Input, (b) DDN, (c) RESCAN, (d) NLEDN, (e) REHEN, (f) PreNet, (g) SpaNet, (h) Ours, (i) GT.
4.4. Evaluation on Network Design

To discuss the design of the PFR-Net, we perform ablation studies on Rain100H dataset by evaluating the benefit of the introduced fully residual learning, compact dense connection, feature pyramid and negative SSIM.

4.4.1. Fully Residual Learning

We evaluate the introduced fully residual learning by comparing it with residual and non-residual learning. Fig. 10 illustrates the architectures of no-residual block, two types of residual blocks and fully residual block with 4 convolution operations. Table 2 presents the results of PFR-Net embedded...
Fig. 9. The results from real-world datasets. From left to right: (a) Input, (b) DDN, (c) RESCAN, (d) NLEDN, (e) REHEN, (f) PreNet, (g) SpaNet, (h) Ours.

Fig. 10. (a) Non-Residual Block. (b) Residual Block-1. (c) Residual Block-2. (d) Fully Residual Block.

Table 2
Analysis on Fully Residual Learning.

<table>
<thead>
<tr>
<th>Metric</th>
<th>$R_1$</th>
<th>$R_2$</th>
<th>$R_3$</th>
<th>$R_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>30.63</td>
<td>30.57</td>
<td>30.50</td>
<td>30.39</td>
</tr>
<tr>
<td>SSIM</td>
<td>0.9292</td>
<td>0.9287</td>
<td>0.9281</td>
<td>0.9274</td>
</tr>
</tbody>
</table>

Table 3
Analysis on Compact Dense Connection.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Compact Dense Connection</th>
<th>Direct Connection</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>27.91</td>
<td>27.16</td>
</tr>
<tr>
<td>SSIM</td>
<td>0.8921</td>
<td>0.8760</td>
</tr>
</tbody>
</table>
with these blocks in the PFR-Units. Due to more identity mappings involved to encourage the feature re-usage, the PFR-Net with fully residual learning gains the highest SSIM and PSRN.

### Table 4
Analysis on Feature Pyramid.

<table>
<thead>
<tr>
<th>Metric</th>
<th>$K = 1$</th>
<th>$K = 2$</th>
<th>$K = 3$</th>
<th>$K = 4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>29.53</td>
<td>30.28</td>
<td>30.38</td>
<td>30.63</td>
</tr>
<tr>
<td>SSIM</td>
<td>0.9174</td>
<td>0.9254</td>
<td>0.9272</td>
<td>0.9292</td>
</tr>
</tbody>
</table>

### Table 5
Results of number of channels. $C$ denotes the number of channels.

<table>
<thead>
<tr>
<th>$C$</th>
<th>PSNR</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$24$</td>
<td>30.37</td>
<td>0.9264</td>
</tr>
<tr>
<td>$28$</td>
<td>30.49</td>
<td>0.9277</td>
</tr>
<tr>
<td>$32$</td>
<td>30.63</td>
<td>0.9292</td>
</tr>
<tr>
<td>$34$</td>
<td>30.55</td>
<td>0.9288</td>
</tr>
<tr>
<td>$36$</td>
<td>30.53</td>
<td>0.9286</td>
</tr>
<tr>
<td>$38$</td>
<td>30.64</td>
<td>0.9298</td>
</tr>
</tbody>
</table>

![PSNR and SSIM curves about the number of channels](a.png)

Fig. 11. The PSNR and SSIM curves about the number of channels.

### Table 6
Results of number of PFR-Units. $N$ denotes the number of PFR-Units.

<table>
<thead>
<tr>
<th>$N$</th>
<th>PSNR</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$16$</td>
<td>30.08</td>
<td>0.9237</td>
</tr>
<tr>
<td>$18$</td>
<td>30.41</td>
<td>0.9271</td>
</tr>
<tr>
<td>$20$</td>
<td>30.63</td>
<td>0.9292</td>
</tr>
<tr>
<td>$22$</td>
<td>30.65</td>
<td>0.9296</td>
</tr>
</tbody>
</table>

![PSNR and SSIM curves about the number of PFR-Units](a.png)

Fig. 12. The PSNR and SSIM curves about the number of PFR-Units.
4.4.2. Compact Dense Connection

We further provide an analysis on the compact dense connection. In the experiment, if PFR-Units are directly connected without the compact dense connection, the PFR-Net cannot be trained due to the weak gradients pass in the deep PFR-Net. We conduct another experiment on a lightweight PFR-Net in which the channel number C, the PFR-Unit number L and the convolution number N in the fully residual block are decreased to 16, 10 and 4, respectively. We present the comparison of results in Table 3, which shows that the compact dense connection improves the performance for the lightweight PFR-Net. In summary, for the PFR-Net with shallow depth, the compact dense connection strengthens the feature propagation and improves the performance of the network. And for the PFR-Net with deep depth, the compact dense connection also improves the gradients pass and makes the network can be trained.

4.4.3. Feature Pyramid

We evaluate the feature pyramid in the de-raining unit. The size of the input patches in the training is set to 100 × 100. According to the pooling operation illustrated in Eq. 11, the maximum number of the level in the feature pyramid is 4. In Table 4, we present the results of our PFR-Net using the feature pyramid with different levels. It is noted that the PFR-Net benefits from the increased levels of the pyramid and achieves the best performance with 4 levels. Compared with the single feature scale, the feature pyramid with 4 levels improves the SSIM and PSRN by 0.12% and 1.1 dB, respectively.

4.4.4. Analysis on the Number of Channels

In this section, we give an analysis about the effect on the number of channels. The results of the number of channels are shown in Table 5 and Fig. 11. As one can see that the results get better as the number of channels when C ≤ 32, while the results get worse when 32 < C ≤ 6. The results gain the best when the C = 38 but the rate of increase is smaller compared with C = 32. So, we select C = 32 as our network settings in terms of the performance and model size as a trade-off.

4.4.5. Analysis on the Number of PFR-Units

We analyze the number of PFR-units for the effect on the deraining results. The results are shown in Table 6 and Fig. 12. We can see that the result are better when N = 20. Although the results are the best when N = 22, the rate of increase is smaller compared with N = 20. Hence, we select N = 20 as the network settings in terms of the deraining performance and model sizes as a trade-off.

4.4.6. Negative SSIM Loss

In Fig. 13, we compare the PFR-Net training with negative SSIM loss and MSE loss. With same number of iterations, using negative SSIM loss, the PFR-Net gains higher SSIM and PSRN. This comparison demonstrates that negative SSIM loss preserves more structural and textural information than MSE loss and our PFR-Net benefits from the negative SSIM loss.

5. Conclusion

In this paper, we design an efficient de-raining unit and propose a deep de-raining convolutional neural network. We employ feature pyramid to learn the rain streaks with different scales and shapes. And we re-use the features fully in the de-raining unit and propagate the features between the de-raining units to make the deep de-raining network easy to learn and improve the performance of the network. Quantitative and qualitative experimental results demonstrate the superiority of the proposed method compared with several state-of-the-art de-raining methods on Rain100H, Rain100L and Rain1200 datasets.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

G. Yao, C. Wang, Y. Wu et al.

Neurocomputing 456 (2021) 168–179
Yang Wang received his master's degree from Nanjing University of Aeronautics and Astronautics in 2017. Currently studying for a PhD in Nanjing University of Aeronautics and Astronautics. His research interest lies in the areas of computer vision.