Progressive Image Deraining Networks: A Better and Simpler Baseline

Dongwei Ren[1], Wangmeng Zuo[2], Qinghua Hu[1], Pengfei Zhu[1], and Deyu Meng[3]

[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
[3]Xi'an Jiaotong University, Xi'an, China

Abstract

Along with the deraining performance improvement of deep networks, their structures and learning become more and more complicated and diverse, making it difficult to analyze the contribution of various network modules when developing new deraining networks. To handle this issue, this paper provides a better and simpler baseline deraining network by considering network architecture, input and output, and loss functions. Specifically, by repeatedly unfolding a shallow ResNet, progressive ResNet (PRN) is proposed to take advantage of recursive computation. A recurrent layer is further introduced to exploit the dependencies of deep features across stages, forming our progressive recurrent network (PReNet). Furthermore, intra-stage recursive computation of ResNet can be adopted in PRN and PReNet to notably reduce network parameters with unsubstantial degradation in deraining performance. For network input and output, we take both stage-wise result and original rainy image as input to each ResNet and finally output the prediction of residual image. As for loss functions, single MSE or negative SSIM losses are sufficient to train PRN and PReNet. Experiments show that PRN and PReNet perform favorably on both synthetic and real rainy images. Considering its simplicity, efficiency and effectiveness, our models are expected to serve as a suitable baseline in future deraining research. The source codes are available at `https://github.com/csdwren/PReNet`.

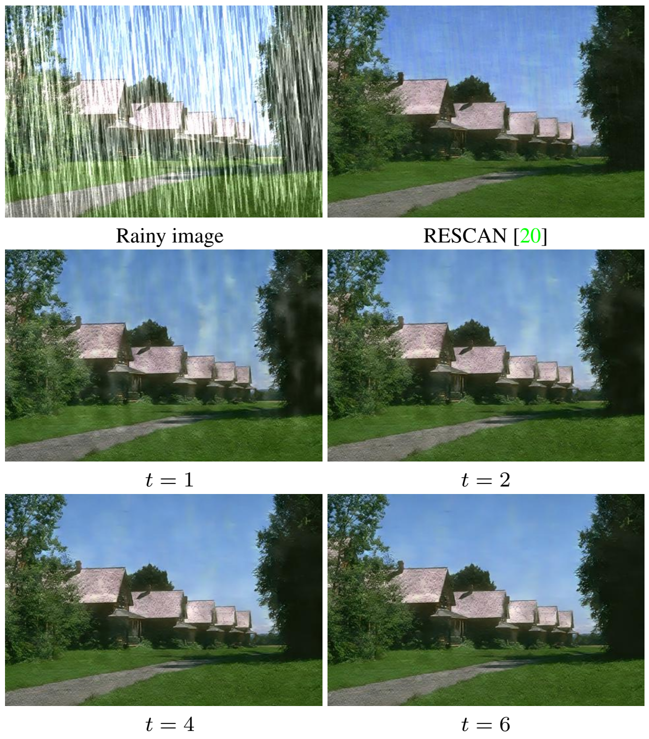
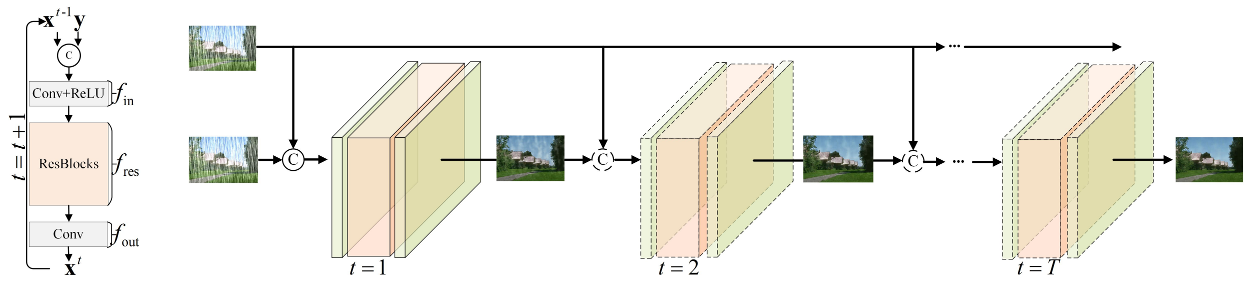


Figure 1. Deraining results by RESCAN [20] and PReNet ($T = 6$) at stage $t = 1, 2, 4, 6$, respectively.

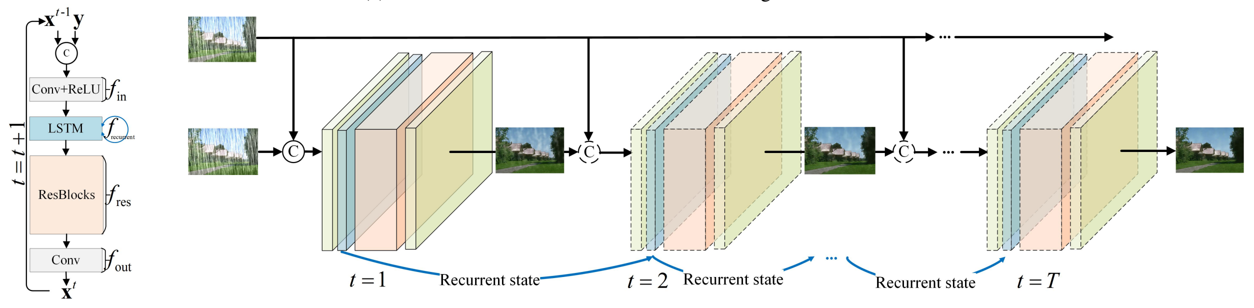
1. Introduction

Rain is a common weather condition, and has severe adverse effect on not only human visual perception but also the performance of various high level vision tasks such as image classification, object detection, and video surveillance [7, 14]. Single image deraining aims at restoring clean background image from a rainy image, and has drawn considerable recent research attention. For example, several traditional optimization based methods [1, 9, 21, 22] have been suggested for modeling and separating rain streaks from background clean image. However, due to the complex composition of rain and background layers, image deraining remains a challenging ill-posed problem.

Driven by the unprecedented success of deep learning in low level vision [3, 15, 18, 28, 34], recent years have also witnessed the rapid progress of deep convolutional neural network (CNN) in image deraining. In [5], Fu *et al.* show that it is difficult to train a CNN to directly predict background


$\mathbf{x}^{t-1}$ $\mathbf{y}$
C
Conv+ReLU f_{in}
ResBlocks f_{res}
Conv f_{out}
$\mathbf{x}^{t}$
$t=t+1$
$t=1$
$t=2$
$t=T$

(a) PRN and the illustration of PRN with T stages recursion


$\mathbf{x}^{t-1}$ $\mathbf{y}$
C
Conv+ReLU f_{in}
LSTM $f_{\text{recurrent}}$
ResBlocks f_{res}
Conv f_{out}
$\mathbf{x}^{t}$
$t=t+1$
$t=1$
Recurrent state
$t=2$
Recurrent state
Recurrent state
$t=T$

(b) PReNet and the illustration of PReNet with T stages recursion

Figure 2. The architectures of progressive networks, where f_{in} is a convolution layer with *ReLU*, f_{res} is *ResBlocks*, f_{out} is a convolution layer, $f_{\text{recurrent}}$ is a convolutional LSTM and © is a *concat* layer. f_{res} can be implemented as conventional ResBlocks or recursive ResBlocks as shown in Fig. 3.

image from rainy image, and utilize a 3-layer CNN to remove rain streaks from high-pass detail layer instead of the input image. Subsequently, other formulations are also introduced, such as residual learning for predicting rain steak layer [20], joint detection and removal of rain streaks [30], and joint rain density estimation and deraining [32].

On the other hand, many modules are suggested to constitute different deraining networks, including residual blocks [6, 10], dilated convolution [30, 31], dense blocks [32], squeeze-and-excitation [20], and recurrent layers [20, 25]. Multi-stream [32] and multi-stage [20] networks are also deployed to capture multi-scale characteristics and to remove heavy rain. Moreover, several models are designed to improve computational efficiency by utilizing lightweight networks in a cascaded scheme [4] or a Laplacian pyramid framework [7], but at the cost of obvious degradation in deraining performance. To sum up, albeit the progress of deraining performance, the structures of deep networks become more and more complicated and diverse. As a result, it is difficult to analyze the contribution of various modules and their combinations, and to develop new models by introducing modules to existing deeper and complex deraining networks.

In this paper, we aim to present a new baseline network for single image deraining to demonstrate that: (i) by combining only a few modules, a better and simpler baseline network can be constructed and achieve noteworthy performance gains over state-of-the-art deeper and complex deraining networks, (ii) unlike [5], the improvement of deraining networks may ease the difficulty of training CNNs to directly recover clean image from rainy image. Moreover, the simplicity of baseline network makes it easier to develop new deraining models by introducing other network modules or modifying the existing ones.

To this end, we consider the network architecture, input and output, and loss functions to form a better and simpler baseline networks. In terms of network architecture, we begin with a basic shallow residual network (ResNet) with five residual blocks (ResBlocks). Then, progressive ResNet (PRN) is introduced by recursively unfolding the ResNet into multiple stages without the increase of model parameters (see Fig. 2(a)). Moreover, a recurrent layer [11, 27] is introduced to exploit the dependencies of deep features across recursive stages to form the PReNet in Fig. 2(b). From Fig. 1, a 6-stage PReNet can remove most rain streaks at the first stage, and then remaining rain streaks can be progressively removed, leading to promising deraining quality at the final stage. Furthermore, PRN$_r$ and PReNet$_r$ are presented by adopting intra-stage recursive unfolding of only one ResBlock, which reduces network parameters only at the cost of unsubstantial performance degradation.

Using PRN and PReNet, we further investigate the effect of network input/output and loss function. In terms of network input, we take both stage-wise result and original rainy image as input to each ResNet, and empirically find that the introduction of original image does benefit deraining performance. In terms of network output, we adopt the residual learning formulation by predicting rain streak layer, and find that it is also feasible to directly learn a PRN or PReNet model for predicting clean background from rainy image. Finally, instead of hybrid losses with careful hyper-parameters tuning [4, 6], a single negative SSIM [29] or MSE loss can readily train PRN and PReNet with favorable deraining performance.

Comprehensive experiments have been conducted to evaluate our baseline networks PRN and PReNet. On four synthetic datasets, our PReNet and PRN are computationally very efficient, and achieve much better quantitative and qualitative deraining results in comparison with the state-of-the-art methods. In particular, on heavy rainy dataset Rain100H [30], the performance gains by our PRN and PReNet are still significant. The visually pleasing deraining results on real rainy images and videos have also validated the generalization ability of the trained PReNet and PRN models.

The contribution of this work is four-fold:

- Baseline deraining networks, *i.e.*, PRN and PReNet, are proposed, by which better and simpler networks can work well in removing rain streaks, and provide a suitable basis to future studies on image deraining.
- By taking advantage of intra-stage recursive computation, PRN$_r$ and PReNet$_r$ are also suggested to reduce network parameters while maintaining state-of-the-art deraining performance.
- Using PRN and PReNet, the deraining performance can be further improved by taking both stage-wise result and original rainy image as input to each ResNet, and our progressive networks can be readily trained with single negative SSIM or MSE loss.
- Extensive experiments show that our baseline networks are computationally very efficient, and perform favorably against state-of-the-arts on both synthetic and real rainy images.

2. Related Work

In this section, we present a brief review on two representative types of deraining methods, *i.e.*, traditional optimization-based and deep network-based ones.

2.1. Optimization-based Deraining Methods

In general, a rainy image can be formed as the composition of a clean background image layer and a rain layer. On the one hand, linear summation is usually adopted as the composition model [1, 21, 35]. Then, image deraining can be formulated by incorporating with proper regularizers on both background image and rain layer, and solved by specific optimization algorithms. Among these methods, Gaussian mixture model (GMM) [21], sparse representation [35], and low rank representation [1] have been adopted for modeling background image or a rain layers. Based on linear summation model, optimization-based methods have been also extended for video deraining [8, 12, 13, 16, 19]. On the other hand, screen blend model [22, 26] is assumed to be more realistic for the composition of rainy image, based on which Luo *et al.* [22] use the discriminative dictionary learning to separate rain streaks by enforcing the two layers share fewest dictionary atoms. However, the real composition generally is more complicated and the regularizers are still insufficient in characterizing background and rain layers, making optimization-based methods remain limited in deraining performance.

2.2. Deep Network-based Deraining Methods

When applied deep network to single image deraining, one natural solution is to learn a direct mapping to predict clean background image $\mathbf{x}$ from rainy image $\mathbf{y}$. However, it is suggested that plain fully convolutional networks (FCN) are ineffective in learning the direct mapping [5, 6]. Instead, Fu *et al.* [5, 6] apply a low-pass filter to decompose $\mathbf{y}$ into a base layer $\mathbf{y}_{\text{base}}$ and a detail layer $\mathbf{y}_{\text{detail}}$. By assuming $\mathbf{y}_{\text{base}} \approx \mathbf{x}_{\text{base}}$, FCNs are then deployed to predict $\mathbf{x}_{\text{detail}}$ from $\mathbf{y}_{\text{detail}}$. In contrast, Li *et al.* [20] adopt the residual learning formulation to predict rain layer $\mathbf{y} - \mathbf{x}$ from $\mathbf{y}$. More complicated learning formulations, such as joint detection and removal of rain streaks [30], and joint rain density estimation and deraining [32], are also suggested. And adversarial loss is also introduced to enhance the texture details of deraining result [25, 33]. In this work, we show that the improvement of deraining networks actually eases the difficulty of learning, and it is also feasible to train PRN and PReNet to learn either direct or residual mapping.

For the architecture of deraining network, Fu *et al.* first adopt a shallow CNN [5] and then a deeper ResNet [6]. In [30], a multi-task CNN architecture is designed for joint detection and removal of rain streaks, in which contextualized dilated convolution and recurrent structure are adopted to handle multi-scale and heavy rain steaks. Subsequently, Zhang *et al.* [32] propose a density aware multi-stream densely connected CNN for joint estimating rain density and removing rain streaks. In [25], attentive-recurrent network is developed for single image raindrop removal. Most recently, Li *et al.* [20] recurrently utilize dilated CNN and squeeze-and-excitation blocks to remove heavy rain streaks. In comparison to these deeper and complex networks, our work incorporates ResNet, recurrent layer and multi-stage recursion to constitute a better, simpler and more efficient deraining network.

Besides, several lightweight networks, *e.g.*, cascaded scheme [4] and Laplacian pyrimid framework [7] are also developed to improve computational efficiency but at the cost of obvious performance degradation. As for PRN and PReNet, we further introduce intra-stage recursive computation to reduce network parameters while maintaining state-of-the-art deraining performance, resulting in our PRN_r and PReNet_r models.

3. Progressive Image Deraining Networks

In this section, progressive image deraining networks are presented by considering network architecture, input and output, and loss functions. To this end, we first describe the general framework of progressive networks as well as input/output, then implement the network modules, and finally discuss the learning objectives of progressive networks.

3.1. Progressive Networks

A simple deep network generally cannot succeed in removing rain streaks from rainy images [5, 6]. Instead of designing deeper and complex networks, we suggest to tackle the deraining problem in multiple stages, where a shallow ResNet is deployed at each stage. One natural multi-stage solution is to stack several sub-networks, which inevitably leads to the increase of network parameters and susceptibility to over-fitting. In comparison, we take advantage of inter-stage recursive computation [15, 20, 28] by requiring multiple stages share the same network parameters. Besides, we can incorporate intra-stage recursive unfolding of only 1 ResBlock to significantly reduce network parameters with graceful degradation in deraining performance.

3.1.1 Progressive Residual Network

We first present a progressive residual network (PRN) as shown in Fig. 2(a). In particular, we adopt a basic ResNet with three parts: (i) a convolution layer f_{in} receives network inputs, (ii) several residual blocks (*ResBlocks*) f_{res} extract deep representation, and (iii) a convolution layer f_{out} outputs deraining results. The inference of PRN at stage t can be formulated as

$$\begin{aligned} \mathbf{x}^{t-0.5} &= f_{\text{in}}(\mathbf{x}^{t-1}, \mathbf{y}), \\ \mathbf{x}^{t} &= f_{\text{out}}(f_{\text{res}}(\mathbf{x}^{t-0.5})), \end{aligned} \tag{1}$$

where f_{in}, f_{res} and f_{out} are stage-invariant, *i.e.*, network parameters are reused across different stages.

We note that f_{in} takes the concatenation of the current estimation $\mathbf{x}^{t-1}$ and rainy image $\mathbf{y}$ as input. In comparison to only $\mathbf{x}^{t-1}$ in [20], the inclusion of $\mathbf{y}$ can further improve the deraining performance. The network output can be the prediction of either rain layer or clean background image. Our empirical study show that, although predicting rain layer performs moderately better, it is also possible to learn progressive networks for predicting background image.

3.1.2 Progressive Recurrent Network

We further introduce a recurrent layer into PRN, by which feature dependencies across stages can be propagated to facilitate rain streak removal, resulting in our progressive recurrent network (PReNet). The only difference between PReNet and PRN is the inclusion of recurrent state $\mathbf{s}^t$,

$$\begin{aligned} \mathbf{x}^{t-0.5} &= f_{\text{in}}(\mathbf{x}^{t-1}, \mathbf{y}), \\ \mathbf{s}^{t} &= f_{\text{recurrent}}(\mathbf{s}^{t-1}, \mathbf{x}^{t-0.5}), \\ \mathbf{x}^{t} &= f_{\text{out}}(f_{\text{res}}(\mathbf{s}^{t})), \end{aligned} \tag{2}$$

where the recurrent layer $f_{\text{recurrent}}$ takes both $\mathbf{x}^{t-0.5}$ and the recurrent state $\mathbf{s}^{t-1}$ as input at stage $t-1$. $f_{\text{recurrent}}$ can be implemented using either convolutional Long Short-Term Memory (LSTM) [11, 27] or convolutional Gated Recurrent Unit (GRU) [2]. In PReNet, we choose LSTM due to its empirical superiority in image deraining.

The architecture of PReNet is shown in Fig. 2(b). By unfolding PReNet with T recursive stages, the deep representation that facilitates rain streak removal are propagated by recurrent states. The deraining results at intermediate stages in Fig. 1 show that the heavy rain streak accumulation can be gradually removed stage-by-stage.

3.2. Network Architectures

We hereby present the network architectures of PRN and PReNet. All the filters are with size 3×3 and padding 1×1. Generally, f_{in} is a 1-layer convolution with ReLU nonlinearity [23], f_{res} includes 5 ResBlocks and f_{out} is also a 1-layer convolution. Due to the concatenation of 3-channel RGB $\mathbf{y}$ and 3-channel RGB $\mathbf{x}^{t-1}$, the convolution in f_{in} has 6 and 32 channels for input and output, respectively. f_{out} takes the output of f_{res} (or $f_{\text{recurrent}}$) with 32 channels as input and outputs 3-channel RGB image for PRN (or PReNet). In $f_{\text{recurrent}}$, all the convolutions in LSTM have 32 input channels and 32 output channels. f_{res} is the key component to extract deep representation for rain streak removal, and we provide two implementations, *i.e.*, conventional ResBlocks shown in Fig. 3(a) and recursive ResBlocks shown in Fig. 3(b).

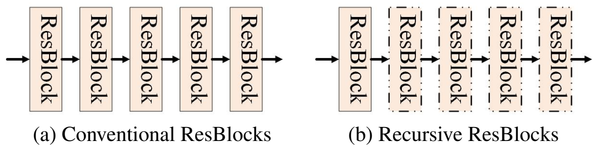


Figure 3. Implementations of f_{res}: (a) conventinal ResBlocks and (b) recursive ResBlocks where one ResBlock is recursively unfolded five times.

Conventional ResBlocks: As shown in Fig. 3(a), we first implement f_{res} with 5 ResBlocks as its conventional form, *i.e.*, each ResBlock includes 2 convolution layers followed by ReLU [23]. All the convolution layers receive 32-channel features without downsampling or upsamping operations. Conventional ResBlocks are adopted in PRN and PReNet.

Recursive ResBlocks: Motivated by [15, 28], we also implement f_{res} by recursively unfolding one ResBlock 5 times, as shown in Fig. 3(b). Since network parameters mainly come from ResBlocks, the intra-stage recursive computation leads to a much smaller model size, resulting in PRN$_r$ and PReNet$_r$. We have evaluated the performance of recursive ResBlocks in Sec. 4.2, indicating its elegant tradeoff between model size and deraining performance.

3.3. Learning Objective

Recently, hybrid loss functions, *e.g.*, MSE+SSIM [4], ℓ_1+SSIM [7] and even adversarial loss [33], have been widely adopted for training deraining networks, but incredibly increase the burden of hyper-parameter tuning. In contrast, benefited from the progressive network architecture, we empirically find that a single loss function, *e.g.*, MSE loss or negative SSIM loss [29], is sufficient to train PRN and PReNet.

For a model with T stages, we have T outputs, *i.e.*, $\mathbf{x}^1$, $\mathbf{x}^2$,..., $\mathbf{x}^T$. By only imposing supervision on the final output $\mathbf{x}^T$, the MSE loss is

$$\mathcal{L} = \|\mathbf{x}^T - \mathbf{x}^{gt}\|^2, \tag{3}$$

and the negative SSIM loss is

$$\mathcal{L} = -\text{SSIM}\left(\mathbf{x}^T, \mathbf{x}^{gt}\right), \tag{4}$$

where $\mathbf{x}^{gt}$ is the corresponding ground-truth clean image. It is worth noting that, our empirical study shows that negative SSIM loss outperforms MSE loss in terms of both PSNR and SSIM.

Moreover, recursive supervision can be imposed on each intermediate result,

$$\mathcal{L} = -\sum\nolimits_{t=1}^{T} \lambda_t \text{SSIM}\left(\mathbf{x}^t, \mathbf{x}^{gt}\right), \tag{5}$$

where λ_t is the tradeoff parameter for stage t. Experimental result in Sec. 4.1.1 shows that recursive supervision cannot achieve any performance gain when $t = T$, but can be adopted to generate visually satisfying result at early stages.

4. Experimental Results

In this section, we first conduct ablation studies to verify the main components of our methods, then quantitatively and qualitatively evaluate progressive networks, and finally assess PReNet on real rainy images and video. All the source code, pre-trained models and results can be found at `https://github.com/csdwren/PReNet`.

Our progressive networks are implemented using Pytorch [24], and are trained on a PC equipped with two NVIDIA GTX 1080Ti GPUs. In our experiments, all the progressive networks share the same training setting. The patch size is 100×100, and the batch size is 18. The ADAM [17] algorithm is adopted to train the models with an initial learning rate 1×10^{-3}, and ends after 100 epochs. When reaching 30, 50 and 80 epochs, the learning rate is decayed by multiplying 0.2.

4.1. Ablation Studies

All the ablation studies are conducted on a heavy rainy dataset [30] with 1,800 rainy images for training and 100 rainy images (Rain100H) for testing. However, we find that 546 rainy images from the 1,800 training samples have the same background contents with testing images. Therefore, we exclude these 546 images from training set, and train all our models on the remaining 1,254 training images.

4.1.1 Loss Functions

Using PReNet ($T = 6$) as an example, we discuss the effect of loss functions on deraining performance, including MSE loss, negative SSIM loss, and recursive supervision loss.

Negative SSIM v.s. MSE. We train two PReNet models by minimizing MSE loss (PReNet-MSE) and negative SSIM loss (PReNet-SSIM), and Table 1 lists their PSNR and SSIM values on Rain100H. Unsurprisingly, PReNet-SSIM outperforms PReNet-MSE in terms of SSIM. We also note that PReNet-SSIM even achieves higher PSNR, partially attributing to that PReNet-MSE may be inclined to get trapped into poor solution. As shown in Fig. 4, the deraining result by PReNet-SSIM is also visually more plausible than that by PReNet-MSE. Therefore, negative SSIM loss is adopted as the default in the following experiments.

Table 1. Comparison of PReNet (T = 6) with different loss functions.

Loss	PReNet-MSE	PReNet-SSIM	PReNet-RecSSIM
PSNR	29.08	29.32	29.12
SSIM	0.880	0.898	0.895

Single v.s. Recursive Supervision. The negative SSIM loss can be imposed only on the final stage (PReNet-SSIM) in Eqn. (4) or recursively on each stage (PReNet-RecSSIM) in Eqn. (5). For PReNet-RecSSIM, we set $\lambda_t = 0.5$ ($t = 1, 2, ..., 5$) and $\lambda_6 = 1.5$, where the tradeoff parameter for the final stage is larger than the others. From Table 1, PReNet-RecSSIM performs moderately inferior to PReNet-SSIM. As shown in Fig. 4, the deraining results by PReNet-SSIM and PReNet-RecSSIM are visually indistinguishable. The results indicate that a single loss on the final stage is sufficient to train progressive networks. Furthermore, Fig. 5 shows the intermediate PSNR and SSIM results at each stage for PReNet-SSIM (T = 6) and PReNet-RecSSIM (T = 6). It can be seen that PReNet-RecSSIM can achieve much better intermediate results than PReNet-SSIM, making PReNet-RecSSIM (T = 6) very promising in comput-

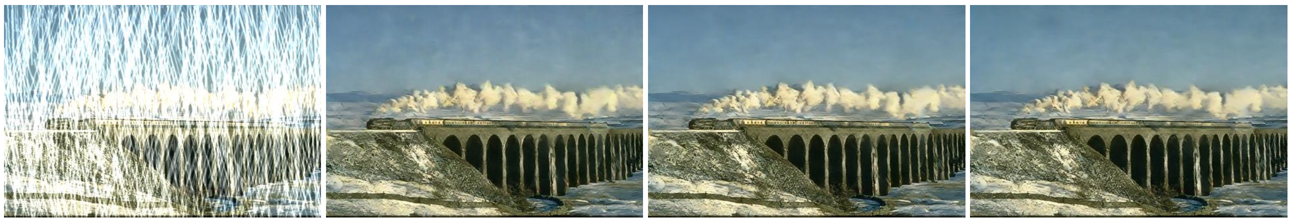


Figure 4. Visual quality comparison of PReNet models trained by different loss functions, including single MSE loss (PReNet-MSE), single negative SSIM loss (PReNet-SSIM) and recursive negative SSIM supervision (PReNet-RecSSIM).

Table 2. Comparison of PReNet models with different T stages.

Model	PReNet$_2$	PReNet$_3$	PReNet$_4$	PReNet$_5$	PReNet$_6$	PReNet$_7$
PSNR	27.27	28.01	28.60	28.92	29.32	29.24
SSIM	0.882	0.885	0.890	0.895	0.898	0.898

Table 3. Comparisons of PReNet variants for ablation studies. PReNet$_x$, PReNet-LSTM, and PReNet-GRU learn direct mapping for predicting background image. In particular, PReNet$_x$ only takes current deraining result $\mathbf{x}^{t-1}$ as input, the recurrent layers in PReNet-LSTM and PReNet-GRU are implemented using LSTM and GRU, respectively. PReNet is the final model by adopting residual learning and LSTM recurrent layer, and taking $\mathbf{y}$ and $\mathbf{x}^{t-1}$ as input.

Model	PReNet$_x$	PReNet-LSTM	PReNet-GRU	PReNet
PSNR	28.91	29.32	29.08	29.46
SSIM	0.895	0.898	0.896	0.899

Table 4. Effect of recursive ResBlocks. PRN and PReNet contain 5 ResBlocks. PRN$_r$ and PReNet$_r$ unfold 1 ResBlock 5 times.

Model	PRN	PReNet	PRN$_r$	PReNet$_r$
PSNR	28.07	29.46	27.43	28.98
SSIM	0.884	0.899	0.874	0.892
#. Parameters	95,107	168,963	21,123	94,979

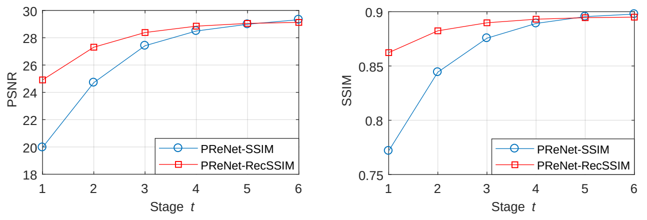


Figure 5. Average PSNR and SSIM of PReNet-SSIM ($T = 6$) and PReNet-RecSSIM ($T = 6$) at stage $t = 1, 2, 3, 4, 5, 6$.

ing resource constrained environments by stopping the inference at any stage t.

4.1.2 Network Architecture

In this subsection, we assess the effect of several key modules of progressive networks, including recurrent layer, multi-stage recursion, and intra-stage recursion.

Recurrent Layer. Using PReNet ($T = 6$), we test two types of recurrent layers, *i.e.*, LSTM (PReNet-LSTM) and GRU (PReNet-GRU). It can be seen from Table 3 that LSTM performs slightly better than GRU in terms of quantitative metrics, and thus is adopted as the default implementation of recurrent layer in our experiments. We further compare progressive networks with and without recurrent layer, *i.e.*, PReNet and PRN, in Table 4, and obviously the introduction of recurrent layer does benefit the deraining performance in terms of PSNR and SSIM.

Intra-stage Recursion. From Table 4, intra-stage recursion, *i.e.*, recursive ResBlocks, is introduced to significantly reduce the number of parameters of progressive networks, resulting in PRN$_r$ and PReNet$_r$. As for deraining performance, it is reasonable to see that PRN and PReNet respectively achieve higher average PSNR and SSIM values than PRN$_r$ and PReNet$_r$. But in terms of visual quality, PRN$_r$ and PReNet$_r$ are comparable with PRN and PReNet, as shown in Fig. 6.

Recursive Stage Number T. Table 2 lists the PSNR and SSIM values of four PReNet models with stages $T = 2, 3, 4, 5, 6, 7$. One can see that PReNet with more stages (from 2 stages to 6 stages) usually leads to higher average PSNR and SSIM values. However, larger T also makes PReNet more difficult to train. When $T = 7$, PReNet$_7$ performs a little inferior to PReNet$_6$. Thus, we set $T = 6$ in the following experiments.

4.1.3 Effect of Network Input/Output

Network Input. We also test a variant of PReNet by only taking $\mathbf{x}^{t-1}$ at each stage as input to f_{in} (*i.e.*, PReNet$_x$), where such strategy has been adopted in [20, 30]. From Table 3, PReNet$_x$ is obviously inferior to PReNet in terms of both PSNR and SSIM, indicating the benefit of receiving $\mathbf{y}$ at each stage.

Network Output. We consider two types of network outputs by incorporating residual learning formulation (*i.e.*,

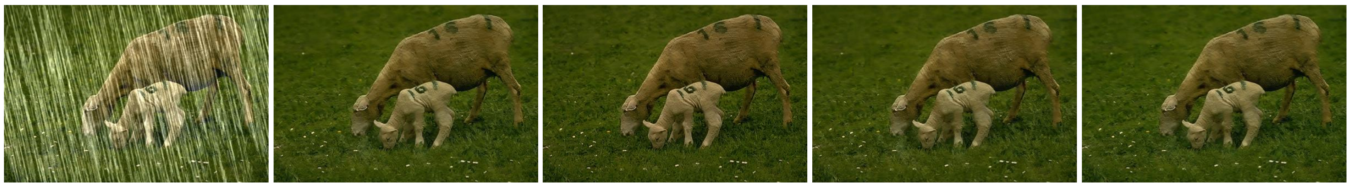

(a) Rainy image (b) PRN (c) PReNet (d)PRN$_r$ (e) PReNet$_r$

Figure 6. Visual effects of recursive ResBlocks. The deraining results by PRN$_r$ and PReNet$_r$ are visually indistinguishable with those by PRN and PReNet.

PReNet in Table 3) or not (*i.e.*, PReNet-LSTM in Table 3). From Table 3, residual learning can make a further contribution to performance gain. It is worth noting that, benefited from progressive networks, it is feasible to learn PReNet for directly predicting clean background from rainy image, and even PReNet-LSTM can achieve appealing deraining performance.

4.2. Evaluation on Synthetic Datasets

Our progressive networks are evaluated on three synthetic datasets, *i.e.*, Rain100H [30], Rain100L [30] and Rain12 [21]. Five competing methods are considered, including one traditional optimization-based method (GMM [21]) and three state-of-the-art deep CNN-based models, *i.e.*, DDN [6], JORDER [30] and RESCAN [20], and one lightweight network (RGN [4]). For heavy rainy images (Rain100H) and light rainy images (Rain100L), the models are respectively trained, and the models for light rain are directly applied on Rain12. Since the source codes of RGN are not available, we adopt the numerical results reported in [4]. As for JORDER, we directly compute average PSNR and SSIM on deraining results provided by the authors. We re-train RESCAN [20] for Rain100H with the default settings. Besides, we have tried to train RESCAN for light rainy images, but the results are much inferior to the others. So its results on Rain100L and Rain12 are not reported in our experiments.

Our PReNet achieves significant PSNR and SSIM gains over all the competing methods. *We also note that for Rain100H, RESCAN [20] is re-trained on the full 1,800 rainy images, the performance gain by our PReNet trained on the strict 1,254 rainy images is still notable.* Moreover, even PReNet$_r$ can perform better than all the competing methods. From Fig. 7, visible dark noises along rain directions can still be observed from the results by the other methods. In comparison, the results by PRN and PReNet are visually more pleasing.

We further evaluate progressive networks on another dataset [6] which includes 12,600 rainy images for training and 1,400 rainy images for testing (Rain1400). From Table 6, all the four versions of progressive networks outperform DDN in terms of PSNR and SSIM. As shown in Fig. 8, the visual quality improvement by our methods is also significant, while the result by DDN still contains visible rain streaks.

Table 7 lists the running time of different methods based on a computer equipped with a NVIDIA GTX 1080Ti GPU. We only give the running time of DDN [6], JORDER [30] and RESCAN [20], due to the codes of the other competing methods are not available. We note that the running time of DDN [6] takes the separation of details layer into account. Unsurprisingly, PRN and PReNet are much more efficient due to its simple network architecture.

4.3. Evaluation on Real Rainy Images

Using two real rainy images in Fig. 9, we compare PReNet with two state-of-the-art deep methods, *i.e.*, JORDER [30] and DDN [6]. It can be seen that PReNet and JORDER perform better than DDN in removing rain streaks. For the first image, rain streaks remain visible in the result by DDN, while PReNet and JORDER can generate satisfying deraining results. For the second image, there are more or less rain streaks in the results by DDN and JORDER, while the result by PReNet is more clear.

4.4. Evaluation on Real Rainy Videos

Finally, PReNet is adopted to process a rainy video in a frame-by-frame manner, and is compared with state-of-the-art video deraining method, *i.e.*, FastDerain [12]. As shown in Fig. 10, for frame #510, both FastDerain and our PReNet can remove all the rain streaks, indicating the performance of PReNet even without the help of temporal consistency. However, FastDerain fails in switching frames, since it is developed by exploiting the consistency of adjacent frames. As a result, for frame #571, #572 and 640, rain streaks are remained in the results by FastDerain, while our PReNet performs favorably and is not affected by switching frames and accumulation error.

5. Conclusion

In this paper, a better and simpler baseline network is presented for single image deraining. Instead of deeper and complex networks, we find that the simple combination of ResNet and multi-stage recursion, *i.e.*, PRN, can result in favorable performance. Moreover, the deraining

Table 5. Average PSNR and SSIM comparison on the synthetic datasets Rain100H [30], Rain100L [30] and Rain12 [21]. Red, blue and cyan colors are used to indicate top 1st, 2nd and 3rd rank, respectively. $^{\triangleright}$ means these metrics are copied from [4]. $^{\circ}$ means the metrics are directly computed based on the deraining images provided by the authors [30]. * donates the method is re-trained with their default settings (*i.e.*, all the 1800 training samples for Rain100H).

Method	GMM [21]	DDN [6]	RGN [4]$^{\triangleright}$	JORDER [30]$^{\circ}$	RESCAN [20]*	PRN	PReNet	PRN$_r$	PReNet$_r$
Rain100H	15.05/0.425	21.92/0.764	25.25/0.841	26.54/0.835	28.88/0.866	28.07/0.884	29.46/0.899	27.43/0.874	28.98/0.892
Rain100L	28.66/0.865	32.16/0.936	33.16/0.963	36.61/0.974	——	36.99/0.977	37.48/0.979	36.11/0.973	37.10/0.977
Rain12	32.02/0.855	31.78/0.900	29.45/0.938	33.92/0.953	——	36.62/0.952	36.66/0.961	36.16/0.961	36.69/0.962

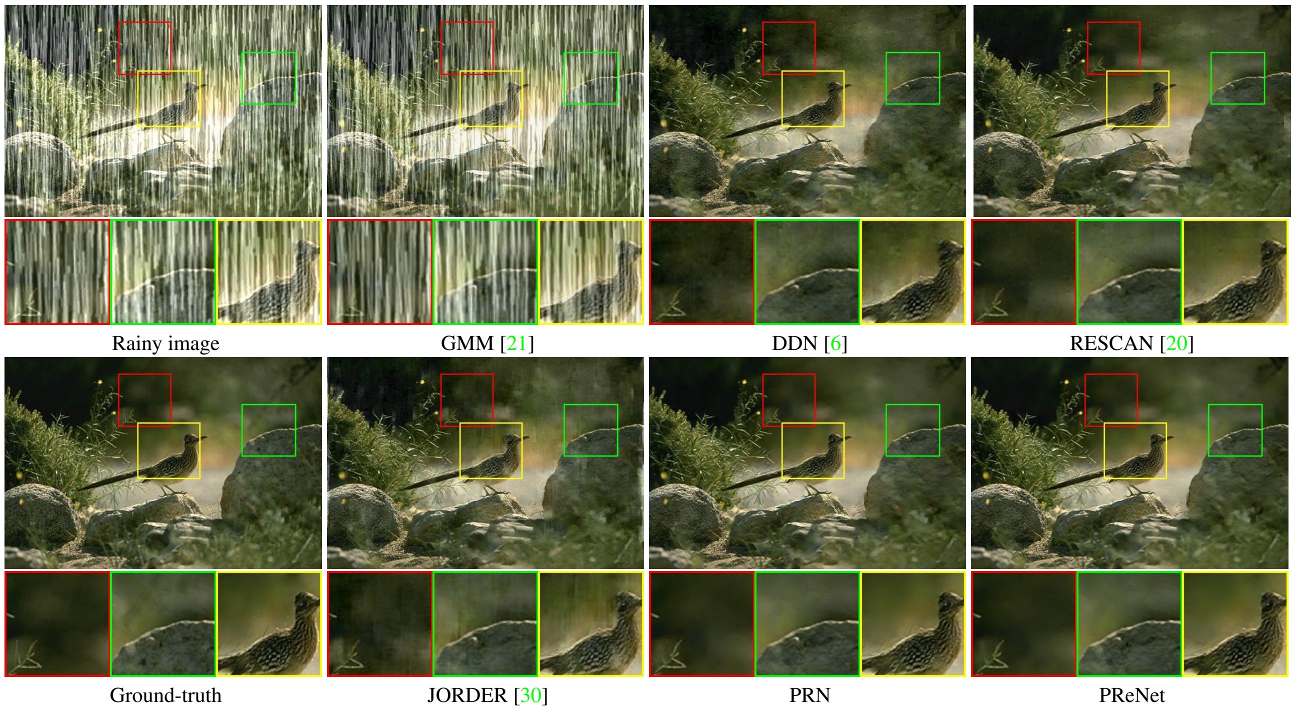


Figure 7. Visual quality comparison on an image from Rain100H [30].

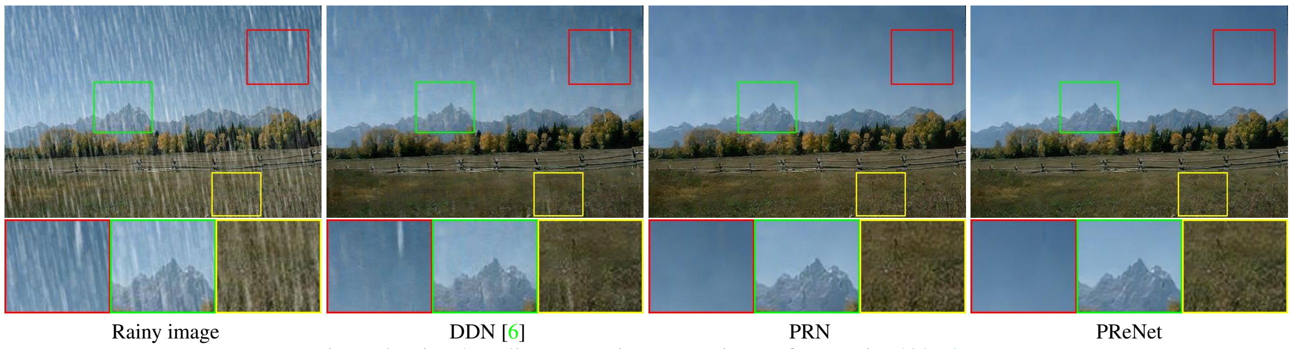


Figure 8. Visual quality comparison on an image from Rain1400 [6].

Table 6. Quantitative comparison on Rain1400 [6].

Method	DDN [6]	PRN	PReNet	PRN$_r$	PReNet$_r$
PSNR	29.91	31.69	32.60	31.31	32.44
SSIM	0.910	0.941	0.946	0.937	0.944

performance can be further boosted by the inclusion of recurrent layer, and stage-wise result is also taken as input to each ResNet, resulting in our PReNet model. Furthermore, the network parameters can be reduced by incorporating inter- and intra-stage recursive computation (PRN$_r$ and PReNet$_r$). And our progressive deraining networks can be readily trained with single negative SSIM or MSE loss. Extensive experiments validate the superiority of our PReNet and PReNet$_r$ on synthetic and real rainy images in comparison to state-of-the-art deraining methods. Taking their simplicity, effectiveness and efficiency into account, it is also

Table 7. Comparison of running time (s)

Image Size	DDN [6]	JORDER [30]	RESCAN [20]	PRN	PReNet
500×500	0.407	0.179	0.448	0.088	0.156
1024×1024	0.754	0.815	1.808	0.296	0.551

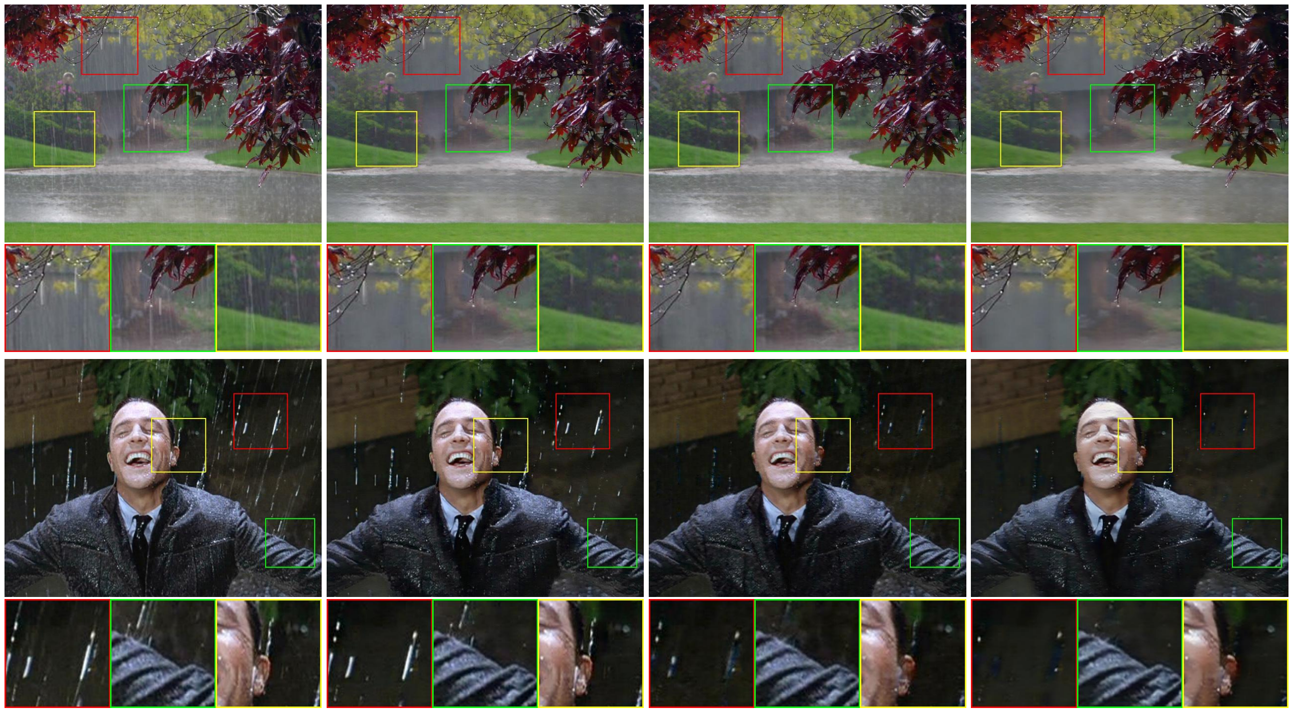


Figure 9. Visual quality comparison on two real rainy images.

appealing to exploit our models as baselines when developing new deraining networks.

References

[1] Y.-L. Chen and C.-T. Hsu. A generalized low-rank appearance model for spatio-temporally correlated rain streaks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1968–1975, 2013. 1, 3

[2] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 4

[3] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016. 1

[4] Z. Fan, H. Wu, X. Fu, Y. Hunag, and X. Ding. Residual-guide feature fusion network for single image deraining. In *ACM Multimedia*, 2018. 2, 3, 5, 7, 8

[5] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley. Clearing the skies: A deep network architecture for single-image rain removal. *IEEE Transactions on Image Processing*, 26(6):2944–2956, 2017. 1, 2, 3, 4

[6] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley. Removing rain from single images via a deep detail network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1715–1723, 2017. 2, 3, 4, 7, 8, 9

[7] X. Fu, B. Liang, Y. Huang, X. Ding, and J. Paisley. Lightweight pyramid networks for image deraining. *arXiv preprint arXiv:1805.06173*, 2018. 1, 2, 3, 5

[8] K. Garg and S. K. Nayar. Detection and removal of rain from videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2004. 3

[9] S. Gu, D. Meng, W. Zuo, and L. Zhang. Joint convolutional analysis and synthesis sparse representation for single image layer separation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1717–1725, 2017. 1

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 2

[11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2, 4

[12] T. Jiang, T. Huang, X. Zhao, L. Deng, and Y. Wang. A novel tensor-based video rain streaks removal approach via utilizing discriminatively intrinsic priors. In *Proceedings of the*

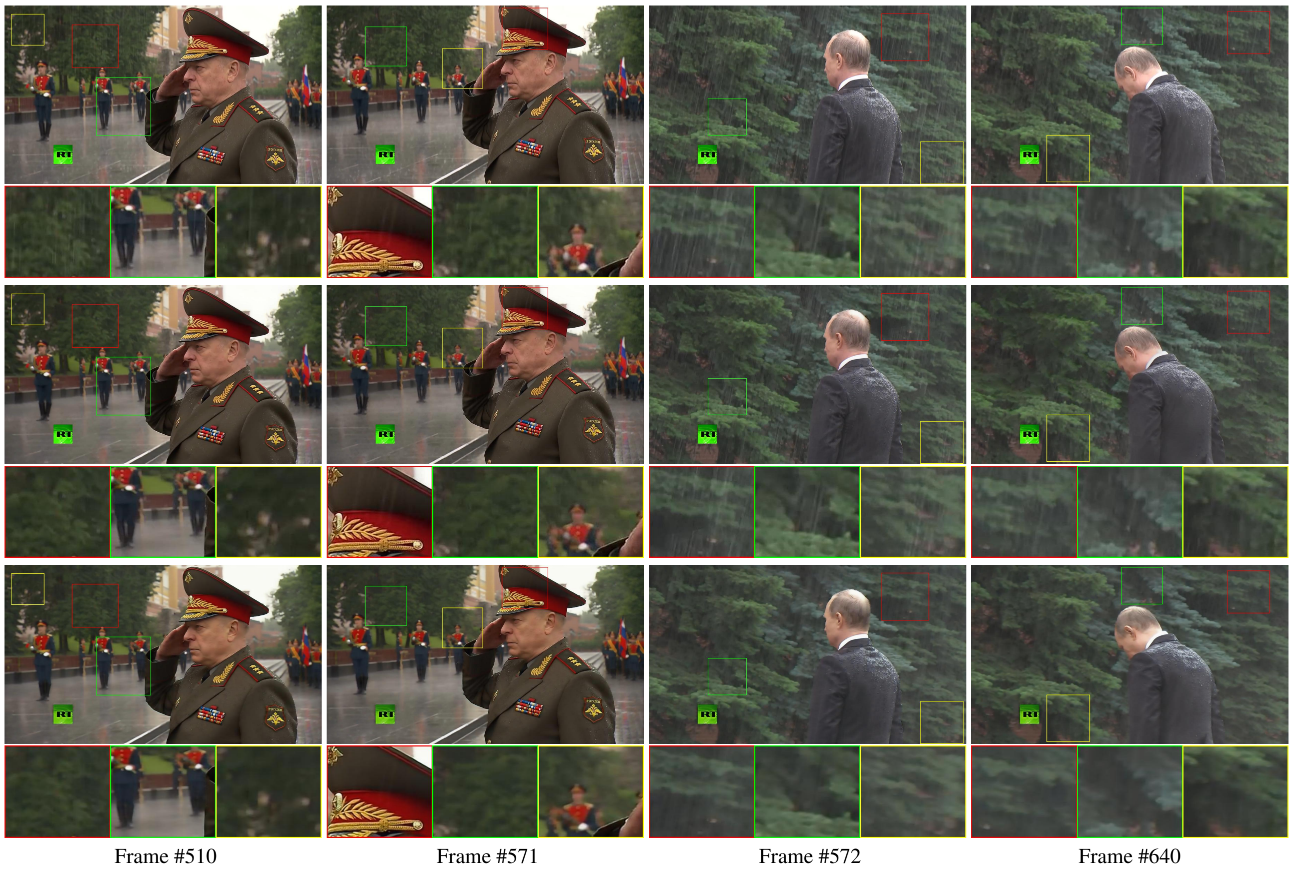


Figure 10. Visual quality comparison on a real rainy video. The first row is rainy frames, the second row is the results by FastDerain [12] and the third row is the results by PReNet.

IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 3, 7, 10

[13] T. Jiang, T. Huang, X. Zhao, L. Deng, and Y. Wang. Fastderain: A novel video rain streak removal method using directional gradient priors. *arXiv preprint arXiv:1803.07487*, 2018. 3

[14] L.-W. Kang, C.-W. Lin, and Y.-H. Fu. Automatic single-image-based rain streaks removal via image decomposition. *IEEE Transactions on Image Processing*, 21(4):1742, 2012. 1

[15] J. Kim, J. Kwon Lee, and K. Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1637–1645, 2016. 1, 4

[16] J.-H. Kim, J.-Y. Sim, and C.-S. Kim. Video deraining and desnowing using temporal correlation and low-rank matrix completion. *IEEE Transactions on Image Processing*, 24(9):2658–2670, 2015. 3

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representation*, 2015. 5

[18] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1

[19] M. Li, Q. Xie, Q. Zhao, W. Wei, S. Gu, J. Tao, and D. Meng. Video rain streak removal by multiscale convolutional sparse coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6644–6653, 2018. 3

[20] X. Li, J. Wu, Z. Lin, H. Liu, and H. Zha. Recurrent squeeze-and-excitation context aggregation net for single image deraining. In *European Conference on Computer Vision*, pages 262–277, 2018. 1, 2, 3, 4, 6, 7, 8, 9

[21] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown. Rain streak removal using layer priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2736–2744, 2016. 1, 3, 7, 8

[22] Y. Luo, Y. Xu, and H. Ji. Removing rain from a single image via discriminative sparse coding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3397–3405, 2015. 1, 3

[23] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the International Conference on Machine Learning*, pages 807–814, 2010. 4

[24] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS Autodiff Workshop:*

The Future of Gradient-based Machine Learning Software and Techniques, 2017. 5

[25] R. Qian, R. T. Tan, W. Yang, J. Su, and J. Liu. Attentive generative adversarial network for raindrop removal from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2482–2491, 2018. 2, 3

[26] D. Ren, W. Zuo, D. Zhang, L. Zhang, and M.-H. Yang. Simultaneous fidelity and regularization learning for image restoration. *arXiv preprint arXiv:1804.04522*, 2018. 3

[27] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015. 2, 4

[28] Y. Tai, J. Yang, and X. Liu. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 5, 2017. 1, 4

[29] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 3, 5

[30] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan. Deep joint rain detection and removal from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1357–1366, 2017. 2, 3, 5, 6, 7, 8, 9

[31] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representation*, 2016. 2

[32] H. Zhang and V. M. Patel. Density-aware single image deraining using a multi-stream dense network. In *Proceedings of the IEEE International Conference on Computer Vision*, 2018. 2, 3

[33] H. Zhang, V. Sindagi, and V. M. Patel. Image de-raining using a conditional generative adversarial network. *arXiv preprint arXiv:1701.05957*, 2017. 3, 5

[34] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017. 1

[35] L. Zhu, C.-W. Fu, D. Lischinski, and P.-A. Heng. Joint bilayer optimization for single-image rain streak removal. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2526–2534, 2017. 3