Single Image Deraining Using Bilateral Recurrent Network

Dongwei Ren, Wei Shang, Pengfei Zhu, Qinghua Hu, Deyu Meng and Wangmeng Zuo

***Abstract*—Single image deraining has received considerable progress based on deep convolutional neural network. Most existing deep deraining methods follow residual learning in image denoising to learn rain streak layer, and perform limited in restoring background image layer. In this work, we first propose a single recurrent network (SRN), where the composition pattern of two layers can be exploited and propagated in multiple stages via LSTM, thereby contributing to the remedy of possible over-removal of rain streaks. This simple SRN is effective not only in learning residual mapping for extracting rain streaks, but also in learning direct mapping for predicting clean background image. Furthermore, we propose bilateral recurrent network (BRN) to allow the interplay between rain streak and background image layers. In particular, two SRNs are coupled to simultaneously exploit these two layers. Instead of naive combination, we propose bilateral LSTMs, which not only can respectively propagate deep features across stages, but also bring the interplay between these two SRNs, which is essential in separating two layers from rainy observation. The experimental results demonstrate that our BRN notably outperforms state-of-the-art deep deraining networks on synthetic datasets quantitatively and qualitatively. The proposed methods also perform more favorably in terms of generalization performance on real-world rainy dataset. All the source code and pre-trained models are available at https://github.com/csdwren/RecDerain.**

***Index Terms*—Image deraining, convolutional neural network, recurrent network, LSTM.**

I. Introduction

REMOVING rain streaks plays an important role in many computer vision applications in rainy outdoor scenes, e.g., surveillance, object detection and recognition [1], [2]. Single image deraining is a very challenging ill-posed problem, and has received considerable research attention in recent years [3]–[9]. Basically, image deraining can be treated as an image decomposition problem, i.e., a rainy image $\mathbf{y}$ should be decomposed into a rain streak layer $\mathbf{r}$ and a clean background image layer $\mathbf{x}$. There are several conventional optimization based deraining methods [4], [10]–[12] by studying the composition pattern of rainy image and designing proper regularization priors.

With the great success of deep learning in low level vision tasks, e.g., denoising [13]–[17], super-resolution [18]–[21], motion deblurring [22], [23] etc, deep convolutional neural network (CNN)-based deraining methods also achieve significant performance improvements against conventional optimization based methods. By assuming the linear degradation model $\mathbf{y} = \mathbf{x} + \mathbf{r}$, most existing deep deraining networks follow residual learning in image denoising [13] to learn rain streak layer. As seminal deep deraining approaches, Fu et al. decompose a rainy image into a base layer and a detail layer, and utilize a 3-layer CNN [3] or a deeper residual network (ResNet) [24] to extract rain streaks from the detail layer. Yang et al. [7] propose a joint rain streak detection and removal framework by using a dilated convolution network. Furthermore, many network architectures, e.g., residual-guide fusion network [6], rain density aware multi-stream dense network [25], squeeze-and-excitation context aggregation network [5], Laplacian pyramid network [1] and spatial attentive network [26] have been proposed to better extract rain streaks. Most recently, unsupervised learning [27] and semi-supervised

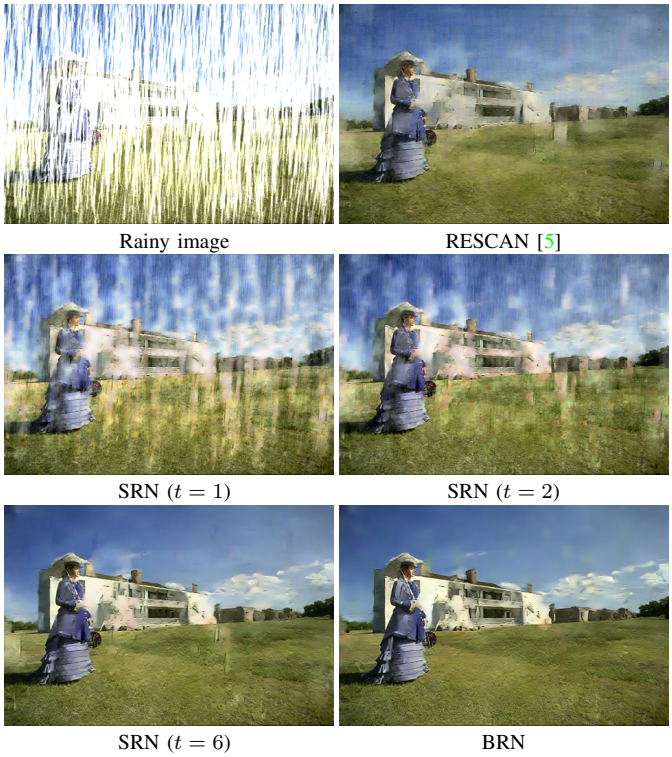

Fig. 1. Deraining results by RESCAN [5], SRN ($T = 6$) at stage $t = 1, 2, 6$, and BRN ($T = 6$), respectively. The result by RESCAN [5] suffers from visible artifacts along with orientations of rain streaks. For SRN, rain streaks can be gradually removed in multiple stages, but the final deraining image still has bright artifacts. The result by BRN is more visually favorable.

D. Ren, W. Shang, P. Zhu and Q. Hu are with the Tianjin Key Laboratory of Machine Learning, College of Intelligence and Computing, Tianjin University, Tianjin, 300350, China. (Email: rendongweihit@gmail.com, shangwei@tju.edu.cn, zhupengfei@tju.edu.cn, huqinghua@tju.edu.cn)

D. Meng is with the National Engineering Laboratory for Algorithm and Analysis Technology on Big Data, Xi'an Jiaotong University, Xi'an, 710049, China. (Email: dymeng@mail.xjtu.edu.cn)

W. Zuo is with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, 150001, China. (Email: cswmzuo@gmail.com)

learning [28] have also been studied for training deraining networks.

Despite of these various network architectures and training strategies, image deraining actually is a layer decomposition task, where both rain streak layer $\mathbf{r}$ and background image layer $\mathbf{x}$ convey rich structural information. For most existing methods, rain streak layer $\mathbf{r}$ is often over-estimated by deep CNN (see Fig. 8 for background textures in estimated rain streak layer), and then clean background image $\mathbf{x}$ is directly predicted by subtracting rain streak layer $\mathbf{r}$ from rainy image $\mathbf{y}$. They ignore the composition and interplay of rain streak and background image layers, and perform limited in restoring clean background image layer. Moreover, the composition pattern of rainy image is much more complicated than linear degradation model [11], [26], and thus clean background image is often over-subtracted, yielding dark artifacts along with the orientations of rain streaks or over-smoothing textures in deraining images (see the results in Figs. 1 and 9).

In this paper, we first propose a single recurrent network (SRN), which is effective in learning direct mapping from rainy image to clean background image. In particular, SRN basically employs a simple shallow ResNet that is recursively unfolded several times to benefit from progressive deraining in multiple stages. Furthermore, a recurrent layer, i.e., convolutional Long Short-Term Memory (LSTM) [29], [30], is introduced into SRN to exploit the dependencies of deep features across recursive stages, as shown in Fig. 2. The progressive model is crucial to remove rain streak layer. In each stage, original rainy image is also taken as input, by which the composition pattern of two layers can be exploited and propagated in intermediate stages via LSTM, thereby contributing to the remedy of possible over-removal of rain streaks. From Fig. 1, one can see that rain streaks can be progressively removed by SRN in multiple stages.

Furthermore, for the layer separation task, it is reasonable that both rain streak and background image layers should be exploited. SRN is able to model either rain streak layer or background image layer. Thus, we further present bilateral recurrent network (BRN) which exploits bilateral LSTMs (BLSTMs) to allow the interplay between these two layers. Specifically, we employ a SRN $\mathcal{F}_r$ to extract rain streak layer $\mathbf{r}$, which is then fed to another SRN $\mathcal{F}_x$ to predict clean background image $\mathbf{x}$, as shown in Fig. 3. Both $\mathcal{F}_r$ and $\mathcal{F}_x$ are recursively unfolded several times. Instead of naive combination, we propose BLSTMs that are incorporated into $\mathcal{F}_r$ and $\mathcal{F}_x$, finally forming our BRN. As shown in Figs. 3 and 4, by adopting BLSTMs, not only the deep features of rain streak layer and background image layer can be respectively propagated across multiple stages, but also the interplay between $\mathcal{F}_r$ and $\mathcal{F}_x$ would facilitate deraining performance. From Fig. 1, one can see that the result by BRN is more visually plausible than those by SRN and RESCAN [5].

In order to train the deraining networks, hybrid loss functions with careful trade-off parameter tuning are usually adopted to achieve good performance [1], [5], [24]. In this paper, our SRN and BRN instead can be easily trained using a single loss function, e.g., mean square error (MSE) loss or negative SSIM loss [31]. Extensive experiments have validated the superiority of BRN against state-of-the-art deep deraining networks [1], [5], [7], [24], [26], [28], [32], [33] on several benchmark datasets. Moreover, our BRN can be well generalized to handle real-world rainy dataset [26], and perform favorably in generating visually plausible deraining results. We summarize our contributions from three aspects:

- We propose SRN to tackle image deraining in a progressive manner, where the composition pattern of rain streak and background image layers can be exploited and propagated in intermediate stages via LSTM.
- We propose BRN for modeling the interplay between rain streak and background image layers, which is also essential in further improving deraining performance.
- Experimental results demonstrate that SRN and BRN achieve notable performance gains against state-of-the-art methods on synthetic and real rainy images.

The remainder of this paper is organized as follows. Section II reviews related works including optimization-based and deep network-based image deraining methods. Section III presents the proposed SRN and BRN for image deraining. Section IV gives the comparison with state-of-the-art methods, and Section V ends this paper with conclusions and discussions.

II. Related Work

In this section, we present a brief review on traditional optimization-based and deep network-based image deraining methods.

A. Optimization-based Deraining Methods

Generally, a rainy image can be represented as the composition of a clean background image layer and a rain streak layer. On the one hand, a linear summation is usually adopted [4], [10], [34], based on which proper regularizers are designed to well pose the deraining problem that is then solved by optimization algorithm. In [10], smoothness regularization and low rank prior are respectively imposed on background image layer and rain streak layer. In [4], two GMMs that are respectively trained on clean image patches and rain streak patches are deployed to model background image layer and rain streak layer. The additive degradation model has also been widely adopted in optimization-based video deraining methods [35]–[39]. On the other hand, it has been suggested that the screen blend model [11] is more realistic for the composition of rainy image. And then the discriminative dictionary learning-based deraining method is proposed [11], where rain streak layer and background image layer are forced to share the fewest dictionary items. However, the composition of real rainy image is much more complicated than existing models, making these optimization-based methods are still facing limited deraining quality.

B. Deep CNN-based Deraining Methods

A natural deraining solution based on deep learning is to adopt CNNs for directly recovering clean background image

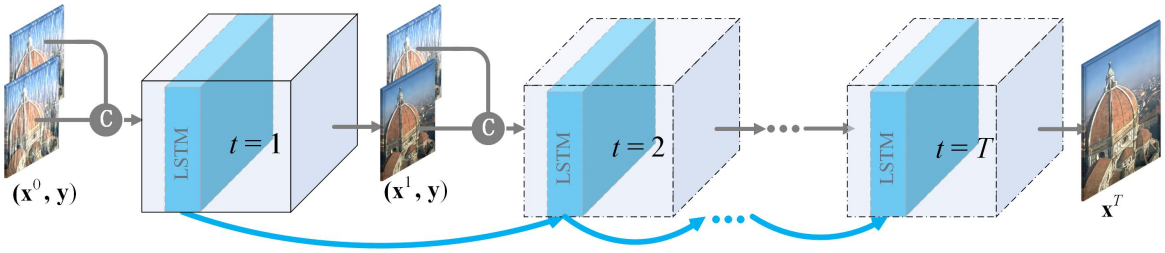

Fig. 2. The illustration of SRN with T stages recursion, where dash lines mean that the network parameters are reused in multiple stages. The architecture details of SRN refer to Eqn. (2).

from rainy image. However, the pioneer works [3], [24] suggest that deep CNNs are ineffective in learning direct mapping from rainy image to background image. Instead, Fu et al. propose to remove rain streaks from the detail layer of a rainy image, where a shallow CNN [3] and a deeper ResNet [24] are successively employed. In [7], a very complicated CNN architecture is designed for jointly detecting and removing rain streaks, in which multi-scale dilation filters are adopted to benefit from larger size receptive field. Li et al. [5] propose to recurrently utilize dilated CNN to handle heavy rain streak accumulation. Zhang et al. [25] propose a density aware multi-stream densely connected CNN for joint estimating rain density and removing rain streaks. Besides, there are several works to incorporate lightweight networks in a cascaded scheme [6] or in a Laplacian pyramid framework [1]. Moreover, in [40], the authors propose to take advantage of adversarial learning to enhance the texture details in derained images. Most recently, unsupervised learning [27] and semi-supervised learning [28] are adopted to train deraining networks. To sump up, despite of various network architectures and training strategies, these deep deraining networks learn residual mapping for extracting rain streaks, which are then directly subtracted from rainy image, often yielding over-subtracted dark artifacts especially for heavy rainy images. In this work, we show that our proposed single recurrent network eases the difficulty of learning, and it is feasible to train SRN for learning either direct mapping or residual mapping. Furthermore, our BRN can simultaneously exploit rain streak layer and clean background image layer, leading to more favorable deraining results.

III. Proposed Method

In this section, we first propose a single recurrent network for direct mapping, which is also effective in residual mapping. Then bilateral recurrent network is proposed by taking advantages of SRN and bilateral LSTMs. Finally, we show that our SRN and BRN models can both be easily trained using a single loss function.

A. Single Recurrent Network

As suggested in [3], [24], a plain deep CNN cannot succeed in directly predicting clean background image from rainy image, and thus deeper and more complicated networks are developed to extract rain streaks via learning residual mapping from rainy image to rain streak layer. However, we suggest that the difficulty of learning deraining network can be eased in multiple stages, and thus it is feasible to train a simple network $\mathcal{F}$ to learn direct mapping from rainy image to clean background image layer. By unfolding $\mathcal{F}$ with T times, the inference at stage t can be formulated as,

$$\mathbf{x}^t = \mathcal{F}(\mathbf{y}, \mathbf{x}^{t-1}), \tag{1}$$

where the parameters of $\mathcal{F}$ are reused across different stages. It is expected that $\mathcal{F}$ can remove rain streaks gradually in each stage.

As shown in Fig. 2, we study $\mathcal{F}$ with four parts: (i) The input layer f_{in} takes the concatenation of rainy image $\mathbf{y}$ and $\mathbf{x}^{t-1}$ from stage $t-1$ as input, (ii) recurrent layer $f_{\text{recurrent}}$ for propagating dependencies across stages to facilitate progressively removing rain streaks, (iii) several residual blocks (*ResBlocks*) f_{res} for extracting deep features, and (iv) output layer f_{out} for generating deraining background image. This simple network is dubbed single recurrent network (SRN) for image deraining. The inference of SRN $\mathcal{F}$ at stage t can be formulated as,

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

where f_{in}, f_{res} and f_{out} are stage-invariant, i.e., network parameters are reused across different stages. The deraining results by SRN in Fig. 1 show that heavy rain streaks accumulation can be gradually removed stage-by-stage. In the following, we present the implementation details of key modules in SRN. We note that all the filters are with size 3×3, padding 1×1 and stride 1.

1) Input Layer f_{in}: Generally, f_{in} is a 1-layer convolution with ReLU nonlinearity [41] for generating deep features $\mathbf{z}^t = f_{\text{in}}(\mathbf{x}^{t-1}, \mathbf{y})$. 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 channels for input, and the output channel number is 32.

2) Recurrent Layer $f_{recurrent}$: At stage t, the recurrent layer $f_{\text{recurrent}}$ receives both the features $\mathbf{z}^t$ from input layer and recurrent state $\mathbf{h}^{t-1}$ from stage $t-1$. $f_{\text{recurrent}}$ can be implemented using either convolutional LSTM [29], [30] or convolutional Gated Recurrent Unit (GRU) [42]. In SRN, we choose LSTM since it performs better quantitatively in experiments for image deraining. The LSTM includes an input

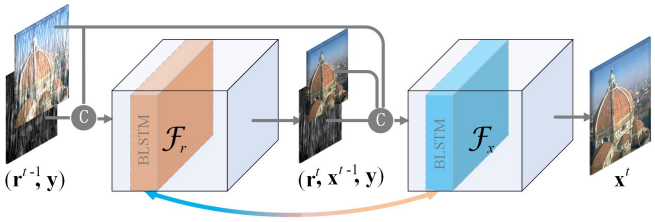

Fig. 3. The architectures of BRN at stage t. BRN consists of two coupled SRNs, i.e., $\mathcal{F}_r$ for extracting rain streaks and $\mathcal{F}_x$ for predicting background image. Instead of two individual LSTMs, BRN can benefit from the interplay between $\mathcal{F}_r$ and $\mathcal{F}_x$ using BLSTMs.

gate $\mathbf{i}^t$, a forget gate $\mathbf{f}^t$, an output gate $\mathbf{o}^t$ and a cell state $\mathbf{c}^t$, and can be formulated as,

$$\begin{aligned}
\mathbf{i}^t &= \sigma(\mathbf{W}_{iz} \otimes \mathbf{z}^t + \mathbf{W}_{is} \otimes \mathbf{h}^{t-1} + \mathbf{b}_i),\\
\mathbf{f}^t &= \sigma(\mathbf{W}_{fz} \otimes \mathbf{z}^t + \mathbf{W}_{fs} \otimes \mathbf{h}^{t-1} + \mathbf{b}_f),\\
\mathbf{o}^t &= \sigma(\mathbf{W}_{oz} \otimes \mathbf{z}^t + \mathbf{W}_{os} \otimes \mathbf{h}^{t-1} + \mathbf{b}_o),\\
\mathbf{g}^t &= tanh(\mathbf{W}_{gz} \otimes \mathbf{z}^t + \mathbf{W}_{gs} \otimes \mathbf{h}^{t-1} + \mathbf{b}_g),\\
\mathbf{c}^t &= \mathbf{f}^t \odot \mathbf{c}^{t-1} + \mathbf{i}^t \odot \mathbf{g}^t,\\
\mathbf{h}^t &= \mathbf{o}^t \odot tanh(\mathbf{c}^t),
\end{aligned} \tag{3}$$

where $\otimes$ is 2D convolution, $\odot$ is entry-wise product, σ is *sigmoid* function, $\mathbf{W}$ and $\mathbf{b}$ are corresponding convolution matrix and bias vector. All the convolutions in LSTM have 32 input channels and 32 output channels.

3) Residual Blocks f_{res}: f_{res} is the key component to extract deep representation for deraining. We implement f_{res} with 5 ResBlocks as its simplest form, i.e., each ResBlock includes 2 convolution layers followed by ReLU [41]. All the convolution layers receive 32-channel features without downsampling or upsamping operations.

4) Output Layer f_{out}: f_{out} takes deep features from f_{res} with 32 channels as input, and generates deraining RGB image with 3 channels.

B. Learning Residual Mapping via SRN

Analogously, SRN can be adopted to learn residual mapping for extracting rain streaks, which can be formulated as,

$$\begin{aligned}
\mathbf{r}^t &= \mathcal{F}(\mathbf{y}, \mathbf{r}^{t-1}),\\
\mathbf{x}^t &= \mathbf{y} - \mathbf{r}^t,
\end{aligned} \tag{4}$$

whre $\mathcal{F}$ shares the same architecture as that in Eqn. (1), except substituting rain streak layer $\mathbf{r}$ as input and output. Due to its narrower mapping range, SRN with residual mapping can achieve a little better results than SRN with direct mapping (refer to the comparison in Table II). Nevertheless, our SRN is more effective in both extracting rain streaks and predicting background image against existing complicated deraining networks.

C. Bilateral Recurrent Network

Although SRN with residual mapping can achieve better quantitative results, its extracted rain streak layer contains some textures from background image (see Fig. 8), and thus the over-subtracted deraining image is likely to be over-smoothed. Taking one step further, we propose to stack two SRNs for extracting rain streaks and predicting clean background image, respectively. At stage t, it can be formulated as,

$$\begin{aligned}
\mathbf{r}^t &= \mathcal{F}_r(\mathbf{y}, \mathbf{r}^{t-1}),\\
\mathbf{x}^t &= \mathcal{F}_x(\mathbf{y}, \mathbf{x}^{t-1}, \mathbf{r}^t),
\end{aligned} \tag{5}$$

where $\mathcal{F}_r$ and $\mathcal{F}_x$ are two coupled SRNs for extracting rain streak layer and clean background image layer, respectively, resulting in coupled recurrent network (CRN) for image deraining. In CRN, two individual LSTMs are adopted to propagate deep features across rain streak layer and background image layer, respectively. We in this work suggest that the interplay between these two layers would facilitate deraining performance. To this end, we first propose bilateral LSTMs (BLSTMs).

1) Bilateral LSTMs: Besides propagating the hidden states across stages, we propose to bring the interplay between these two LSTMs, forming BLSTMs. As shown in Fig. 4, the hidden state $\mathbf{h}_r$ in layer $\mathbf{r}$ is not only propagated across stages to facilitate rain streak extraction, but also is fed to the BLSTM in layer $\mathbf{x}$, and vice versa.

In particular, BLSTM in $\mathcal{F}_r$ at stage t receives the features $\mathbf{z}_r^t = f_r(\mathbf{y}, \mathbf{r}^{t-1})$ from input layer $f_r(\cdot)$, recurrent state $\mathbf{h}_r^{t-1}$ in rain streak layer and recurrent state $\mathbf{h}_x^{t-1}$ from clean background image layer as the input. And then BLSTM in $\mathcal{F}_r$ can be formally expressed as,

$$\begin{aligned}
\mathbf{i}_r^t &= \sigma(\mathbf{W}_{riz} \otimes \mathbf{z}_r^t + \mathbf{W}_{rih_x} \otimes \mathbf{h}_x^{t-1} + \mathbf{W}_{rih_r} \otimes \mathbf{h}_r^{t-1} + \mathbf{b}_{ri}),\\
\mathbf{f}_r^t &= \sigma(\mathbf{W}_{rfz} \otimes \mathbf{z}_r^t + \mathbf{W}_{rfh_x} \otimes \mathbf{h}_x^{t-1} + \mathbf{W}_{rfh_r} \otimes \mathbf{h}_r^{t-1} + \mathbf{b}_{rf}),\\
\mathbf{o}_r^t &= \sigma(\mathbf{W}_{roz} \otimes \mathbf{z}_r^t + \mathbf{W}_{roh_x} \otimes \mathbf{h}_x^{t-1} + \mathbf{W}_{roh_r} \otimes \mathbf{h}_r^{t-1} + \mathbf{b}_{ro}),\\
\mathbf{g}_r^t &= tanh(\mathbf{W}_{rgz} \otimes \mathbf{z}_r^t + \mathbf{W}_{rgh_x} \otimes \mathbf{h}_x^{t-1} + \mathbf{W}_{rgh_r} \otimes \mathbf{h}_r^{t-1} + \mathbf{b}_{rg}),\\
\mathbf{c}_r^t &= \mathbf{f}_r^t \odot \mathbf{c}_r^{t-1} + \mathbf{i}_r^t \odot \mathbf{g}_r^t,\\
\mathbf{h}_r^t &= \mathbf{o}_r^t \odot tanh(\mathbf{c}_r^t).
\end{aligned} \tag{6}$$

BLSTM in $\mathcal{F}_x$ has almost the same architecture with that in $\mathcal{F}_r$ by substituting subscripts r and x. And the minor distinctions are two-fold: (i) $\mathbf{z}_x^t = f_x(\mathbf{y}, \mathbf{x}^{t-1}, \mathbf{r}^t)$ is received as input, and (ii) recurrent state $\mathbf{h}_r^t$ instead of $\mathbf{h}_r^{t-1}$ is taken into account. And the BLSTM in $\mathcal{F}_x$ is formulated as,

$$\begin{aligned}
\mathbf{i}_x^t &= \sigma(\mathbf{W}_{xiz} \otimes \mathbf{z}_x^t + \mathbf{W}_{xih_r} \otimes \mathbf{h}_r^t + \mathbf{W}_{xih_x} \otimes \mathbf{h}_x^{t-1} + \mathbf{b}_{xi}),\\
\mathbf{f}_x^t &= \sigma(\mathbf{W}_{xfz} \otimes \mathbf{z}_x^t + \mathbf{W}_{xfh_r} \otimes \mathbf{h}_r^t + \mathbf{W}_{xfh_x} \otimes \mathbf{h}_x^{t-1} + \mathbf{b}_{xf}),\\
\mathbf{o}_x^t &= \sigma(\mathbf{W}_{xoz} \otimes \mathbf{z}_x^t + \mathbf{W}_{xoh_r} \otimes \mathbf{h}_r^t + \mathbf{W}_{xoh_x} \otimes \mathbf{h}_x^{t-1} + \mathbf{b}_{xo}),\\
\mathbf{g}_x^t &= tanh(\mathbf{W}_{xgz} \otimes \mathbf{z}_x^t + \mathbf{W}_{xgh_r} \otimes \mathbf{h}_r^t + \mathbf{W}_{xgh_x} \otimes \mathbf{h}_x^{t-1} + \mathbf{b}_{xg}),\\
\mathbf{c}_x^t &= \mathbf{f}_x^t \odot \mathbf{c}_x^{t-1} + \mathbf{i}_x^t \odot \mathbf{g}_x^t,\\
\mathbf{h}_x^t &= \mathbf{o}_x^t \odot tanh(\mathbf{c}_x^t).
\end{aligned} \tag{7}$$

By adopting BLSTMs, deep features can be propagated between two layers and across multiple stages.

2) Bilateral Recurrent Network: As shown in Fig. 3, two SRNs $\mathcal{F}_r$ and $\mathcal{F}_x$ can be recursively unfolded T times. BLSTM is incorporated into $\mathcal{F}_r$ and $\mathcal{F}_x$ to respectively propagate deep features of $\mathcal{F}_r$ and $\mathcal{F}_x$ across stages and bring the interplay between $\mathcal{F}_r$ and $\mathcal{F}_x$, forming BRN. At stage t, BRN can be formulated as,

$$\begin{aligned}
\mathbf{r}^t &= \mathcal{F}_r(\mathbf{y}, \mathbf{r}^{t-1}, (\mathbf{h}_r^{t-1}, \mathbf{h}_x^{t-1})),\\
\mathbf{x}^t &= \mathcal{F}_x(\mathbf{y}, \mathbf{x}^{t-1}, \mathbf{r}^t, (\mathbf{h}_r^t, \mathbf{h}_x^{t-1})),
\end{aligned} \tag{8}$$

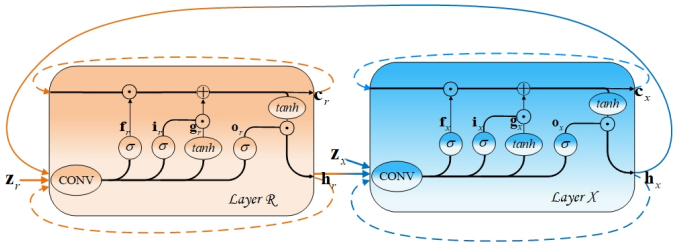

Fig. 4. The architecture of BLSTMs, where the hidden states $\mathbf{h}_r$ and $\mathbf{h}_x$ are not only respectively propagated through layer $\mathcal{X}$ and layer $\mathcal{R}$ (dash lines), but also bring the interplay between layer $\mathcal{R}$ and layer $\mathcal{X}$ (solid lines). The notations are defined in Eqns. (6) and (7). We note that *CONV* here denotes corresponding convolutional matrices and bias vectors.

where $\mathbf{h}_r$ and $\mathbf{h}_x$ are recurrent states for rain streak layer and background image layer in BLSTMs, respectively.

Generally, $\mathcal{F}_r$ and $\mathcal{F}_x$ share the similar SRN architecture. $\mathcal{F}_r$ includes 1 input layer, BLSTM, 3 ResBlocks and 1 output layer. The input layer includes 1 convolution layer and ReLU, and receive the concatenation of rainy image $\mathbf{y}$ and previous rain streak layer $\mathbf{r}^{t-1}$ as the input. The only distinctions of $\mathcal{F}_x$ with $\mathcal{F}_r$ are two-fold: (i) 5 ResBlocks are adopted in $\mathcal{F}_x$, since the background images are usually with richer structures and textures. (ii) The input layer receives not only the concatenation of the rainy image $\mathbf{y}$ and the estimated background image $\mathbf{x}^{t-1}$, but also the current rain streak layer $\mathbf{r}^t$. By simultaneously considering rain streak layer and background image layer, the composition pattern of rainy images can be implicitly learned by $\mathcal{F}_x$. The other parameters including convolution channels, kernel size, padding and stride have the same settings with SRN.

D. Training SRN and BRN

As for training SRN with T stages, we have T outputs, i.e., $\mathbf{x}^1$, $\mathbf{x}^2$,..., $\mathbf{x}^T$, and it is natural to impose recursive supervision on every intermediate result,

$$\mathcal{L} = \sum\nolimits_{t=1}^{T} \lambda_t \ell\left(\mathbf{x}^t, \mathbf{x}^{gt}\right), \tag{9}$$

where $\mathbf{x}^{gt}$ is the corresponding ground-truth clean image, $\ell(\cdot,\cdot)$ measures the difference between the output of SRN at stage t and the corresponding ground-truth, and λ_t is a trade-off parameter.

For BRN with T stages, we have T estimated background images and T rain streak layers, i.e., $\mathbf{x}^1$, $\mathbf{x}^2$,..., $\mathbf{x}^T$ and $\mathbf{r}^1$, $\mathbf{r}^2$,..., $\mathbf{r}^T$. Similar to Eqn (9), the recursive supervision on background image layer and rain streak layer can be defined as,

$$\mathcal{L}_x = \sum\nolimits_{t}^{T} \lambda_t \ell\left(\mathbf{x}^t, \mathbf{x}^{gt}\right), \tag{10}$$

$$\mathcal{L}_r = \sum\nolimits_{t=1}^{T} \lambda_t \ell\left(\mathbf{r}^t, \mathbf{r}^{gt}\right). \tag{11}$$

Then $\mathcal{L}_r$ and $\mathcal{L}_x$ should be balanced by two trade-off parameters α and β,

$$\mathcal{L} = \alpha\mathcal{L}_r + \beta\mathcal{L}_x. \tag{12}$$

Moreover, it is worth noting that we have tried to train SRN and BRN by only imposing supervision on the final output $\mathbf{x}^T$,

$$\mathcal{L} = \ell(\mathbf{x}^T, \mathbf{x}^{gt}). \tag{13}$$

It is interesting to see that the single loss function can lead to better performance for both SRN and BRN (see the results in Sec. IV-A1).

As for the choice of $\ell(\cdot,\cdot)$, several hybrid loss functions, e.g., MSE+SSIM [6], ℓ_1+SSIM [1] or adversarial loss [40], have been widely adopted for training deraining networks. Nevertheless, we empirically find that a single loss function, e.g., MSE loss or negative SSIM loss [31], is sufficient to train both SRN and BRN, which can be attributed that our proposed recurrent network architecture eases the difficulty of learning. Given an image $\mathbf{a}$ and its corresponding ground-truth image $\mathbf{a}^{gt}$, the negative SSIM loss and MSE loss can be respectively formulated as,

$$\ell(\mathbf{a}, \mathbf{a}^{gt}) = -\text{SSIM}\left(\mathbf{a}, \mathbf{a}^{gt}\right), \tag{14}$$

$$\ell(\mathbf{a}, \mathbf{a}^{gt}) = \left\|\mathbf{a} - \mathbf{a}^{gt}\right\|^2. \tag{15}$$

And the experimental results in Sec. IV-A1 indicate that a single negative SSIM loss can achieve superior results quantitatively and qualitatively.

IV. Experimental Results

In this section, we first conduct ablation studies to verify the key contributions of SRN and BRN, then evaluate SRN and BRN against state-of-the-art deraining methods on several benchmark datasets quantitatively and qualitatively, and finally validate their generalization performance on real-world rainy images. More results can be found at https://github.com/csdwren/RecDerain.

Our SRN and BRN are implemented using Pytorch [43], and are trained on a PC equipped with two NVIDIA GTX 1080Ti GPUs. In our experiments, SRN and BRN share the same training settings except specific declarations. The patch size is 100×100. The batch size is 18 for SRN and is 12 for BRN. The ADAM [44] 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.

A. Ablation Study

All the ablation studies are conducted on a heavy rainy dataset [7] with 1,800 rainy images for training and 100 rainy images (Rain100H) for testing. The training and testing images for Rain100H have been updated recently on their project website. However, the reported results in recent deraining works are still based on the original dataset. To keep the consistent comparison, we thus adopt the original dataset for evaluation, where we strictly exclude 546 rainy images from the 1,800 training samples since they have the same background contents with testing images. The performance is quantitatively evaluated using two popular metrics in low-level vision tasks, i.e., SSIM [31] and PSNR.

For both SRN and BRN, the trade-off parameters $\{\lambda_t\}_{t=1}^{T}$ are set as $\lambda_t = 0.5$ $(t = 1, 2, ..., T-1)$ and $\lambda_T = 1.5$, where the trade-off parameter for the final stage is larger than the others. For BRN, the trade-off parameters α = 0.45 and β = 0.55 are adopted. Moreover, the stage number $T = 6$ is set for SRN and BRN except in the subsection for discussing stage number selection.

1) Loss Functions: We discuss the effects of single loss and recursive loss, and the comparison of single MSE loss and single negative SSIM loss.

Single Loss *vs*. Recursive Loss. By taking negative SSIM loss Eqn. (14), SRN and BRN are trained using single SSIM loss (SRN-SSIM and BRN-SSIM) and recursive SSIM loss (SRN-RecSSIM and BRN-RecSSIM), respectively. From Table I, it is interesting to see that SRN-RecSSIM and BRN-RecSSIM perform moderately inferior to SRN-SSIM and BRN-SSIM, respectively. The results indicate that a single loss on the final stage is sufficient to train our SRN and BRN.

MSE Loss *vs*. Negative SSIM Loss. Then we discuss the performance of negative SSIM loss against MSE loss (SRN-MSE and BRN-MSE) under the single supervision. From Table I, one can see that SRN-SSIM and BRN-SSIM are better in terms of not only SSIM metrics but also PSNR metrics. We empirically suggest that a single negative SSIM loss is sufficient to train our SRN and BRN models.

TABLE I
COMPARISON OF SRN AND BRN MODELS WITH DIFFERENT LOSS FUNCTIONS.

	SRN-MSE	SRN-SSIM	SRN-RecSSIM
PSNR	29.08	29.32	29.12
SSIM	0.880	0.898	0.895

(a) Comparison of SRN models with different loss functions.

	BRN-MSE	BRN-SSIM	BRN-RecSSIM
PSNR	29.40	30.47	30.31
SSIM	0.885	0.913	0.908

(b) Comparison of BRN models with different loss functions.

2) Network Architecture of SRN: In this subsection, we assess the contributions of several key modules of SRN, including recurrent layer, network input, and provide the comparison of residual mapping and direct mapping.

Recurrent Layer. In SRN, we test two types of recurrent layers, i.e., LSTM (SRN-LSTM) and GRU (SRN-GRU). It can be seen from Table II 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 methods.

TABLE II
COMPARISONS OF SRN VARIANTS FOR ABLATION STUDIES. SRN_x ONLY ADOPTS $\mathbf{x}^{t-1}$ AS INPUT. SRN-LSTM AND SRN-GRU LEARN DIRECT MAPPING FOR PREDICTING BACKGROUND IMAGE, WHERE LSTM RECURRENT LAYER IS USED IN SRN-LSTM AND GRU RECURRENT LAYER IS USED IN SRN-GRU. SRN IS THE FINAL MODEL BY ADOPTING RESIDUAL MAPPING, LSTM RECURRENT LAYER AND THE CONCATENATION OF $\mathbf{y}$ AND $\mathbf{x}^{t-1}$ AS INPUT.

	SRN_x	SRN-LSTM	SRN-GRU	SRN
PSNR	28.91	29.32	29.08	29.46
SSIM	0.895	0.898	0.896	0.899

Network Input. We test a variant of SRN by only taking $\mathbf{x}^{t-1}$ at each stage as input to f_{in} (i.e., SRN_x), where such strategy has been adopted in [5], [7]. From Table II, SRN_x is obviously inferior to SRN in terms of both PSNR and SSIM, indicating the benefit of receiving $\mathbf{y}$ at each stage.

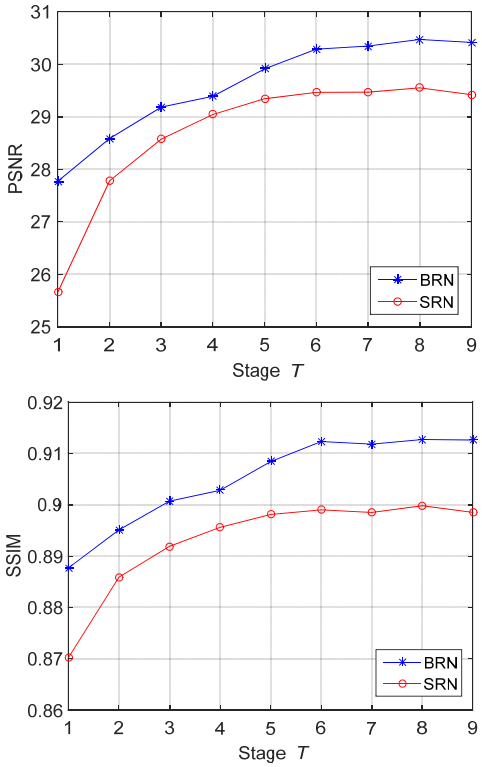

Fig. 6. Average PSNR and SSIM of SRN and BRN with recurrent stage number $T = 1, 2, 3, ..., 9$.

Direct Mapping *vs*. Residual Mapping. By adopting the same SRN architecture, we use SRN to learn direct mapping (i.e., SRN-LSTM in Table II) for predicting background image and to learn residual mapping (i.e., SRN in Table II) for extracting rain streaks. From Table II, SRN is effective in directly generating clean background image, and residual mapping can make a further contribution to performance gain. To sum up, benefited from the recurrent networks, it is feasible to learn SRN in either residual mapping or direct mapping for image deraining, both leading to appealing deraining performance.

3) Network Architecture of BRN: Based on the observations in SRN, we adopt LSTM and BLSTMs in CRN and BRN. To validate the effectiveness of BLSTMs in BRN, we provide several variants of BRN, including CRN (with two individual BLSTMs) and two BLSTMs variants, i.e., $\text{BRN}_{r\to x}$ (information only from $\mathcal{F}_r$ to $\mathcal{F}_x$) and $\text{BRN}_{x\to r}$ (information only from $\mathcal{F}_x$ to $\mathcal{F}_r$). From Table III, one can see that BRN can achieve higher average PSNR and SSIM metrics than CRN, $\text{BRN}_{r\to x}$ and $\text{BRN}_{x\to r}$. It is worth noting that $\text{BRN}_{r\to x}$ is inferior to CRN, and it is owing to that we only supervise the final stage of $\mathbf{x}$, possibly leading to inaccurate rain streaks $\mathbf{r}$. And thus the monodirectional information from $\mathcal{F}_r$ to $\mathcal{F}_x$ would damage the performance. But the interplay between $\mathcal{F}_r$ and $\mathcal{F}_x$ can make significant improvements against all the three variants, indicating the effectiveness of BLSTMs. We also report the visual quality comparison of these variants in Fig.5.

4) Variants of SRN and BRN: We present the comparison of variants of SRN and BRN.

Bilateral GRUs *vs*. Bilateral LSTMs. By adopting the same training settings with BRN, a variant BRN-GRU is trained

Rainy image CRN $\text{BRN}_{r\to x}$ $\text{BRN}_{x\to r}$ BRN

Fig. 5. Visual comparison of BRN variants by adopting different BLSTMs.

TABLE III
EFFECTIVENESS OF BLSTMS IN BRN.

	CRN	$\text{BRN}_{x\to r}$	$\text{BRN}_{r\to x}$	BRN
PSNR	30.04	30.22	30.01	30.47
SSIM	0.909	0.910	0.908	0.913

to verify the effectiveness of bilateral GRUs. In BRN-GRU, LSTMs in both $\mathcal{F}_r$ and $\mathcal{F}_x$ are substituted as GRUs [42]. On Rain100H dataset [7], BRN with bilateral LSTMs outperforms BRN-GRU in terms of both average PSNR and SSIM values, as reported in Table IV. The PSNR decrease ∼0.4dB by BRN-GRU is notable, which can also be verified in terms of visual quality as shown in Fig. 7. Therefore, bilateral LSTMs is a better choice against bilateral GRU for BRN.

Bilateral Recurrent Architecture *vs*. Single Recurrent Architecture. One may argue that SRN with more parameters would perform better than BRN. As reported in Table IV, by increasing ResBlocks number to 18, SRN^+ has the larger number of parameters than BRN and BRN-GRU. SRN^+ indeed achieves better performance than SRN, but it is significantly inferior to BRN. It is worth noting that BRN-GRU has 20% less amount of parameters than SRN^+, but it still performs better than SRN^+ in terms of both PSNR and SSIM values. Thus, the superiority of BRN should be mainly ascribed to the bilateral recurrent architecture instead of simply increasing the number of parameters.

TABLE IV
COMPARISON OF VARIANTS OF SRN AND BRN ON RAIN100H [7]. SRN^+ IS A VARIANT OF SRN BY INCREASING RESBLOCKS NUMBER TO 18. BRN-GRU IS A VARIANT OF BRN BY SUBSTITUTING LSTMS WITH GRUS.

	SRN	SRN^+	BRN-GRU	BRN
PSNR	29.32	29.99	30.08	30.47
SSIM	0.898	0.906	0.908	0.913
#Parameters	168,963	409,411	320,166	375,526

5) Recurrent Stage Number T: We hereby discuss the stage number T for SRN and BRN. Fig. 6 demonstrates PSNR and SSIM values of SRN and BRN models by setting recurrent stage number $T = 1, 2, 3, ..., 9$ trained using single negative SSIM loss. One can see that SRN and BRN have the consistent performance increases along with increasing stage number T from 1 to 6. When $T > 6$, we can observe the performance drops or perturbations for SRN and BRN. Especially, $T = 9$ yields performance drops for both SRN and BRN, possibly due to too long backward propagation distance. Thus, in the following experiments, we set $T = 6$ for both SRN and BRN in comparison with state-of-the-art algorithms.

TABLE VI
QUANTITATIVE COMPARISON ON RAIN1400 DATASET [24].

	DDN [24]	SIRR [28]	BRN
PSNR	29.91	28.44	32.75
SSIM	0.910	0.889	0.948

B. Comparison with State-of-the-arts on Synthetic Datasets

We first evaluate BRN on three benchmark datasets, i.e., Rain100H [7], Rain100L [7] and Rain12 [4], and compare it with several state-of-the-art deraining methods including conventional optimization-based method: GMM [4], state-of-the-art supervised deep CNN-based methods: DDN [24], ResGuideNet [6], JORDER [7] and RESCAN [5], and semi-supervised deep deraining method: SIRR [28]. For heavy rainy images (Rain100H) and light rainy images (Rain100L), the models are respectively trained, and the models for light rain are used to process Rain12. Since the source codes of ResGuideNet are not available, we borrow the quantitative results from [6]. As for JORDER, we directly calculate average PSNR and SSIM on deraining results provided by the authors. As for semi-supervised SIRR [28], we adopt their pre-trained model where SIRR model is fine-tuned using real-world rainy images. It is worth noting that RESCAN and BRN are trained on strict 1,254 rainy images for Rain100H. The comparison results are reported in Table V, from which we can see that our BRN is notably superior to the existing methods. Especially for Rain100H with heavy rain streaks, the performance gain by our BRN is very significant. The semi-supervised SIRR is moderately inferior to the other supervised deraining methods, because real-world rainy images are quite different from these synthetic datasets. As shown in Fig.8, BRN can effectively extract rain streak layer $\mathbf{r}$ using $\mathcal{F}_r$, while the rain streak layers estimated by RESCAN has background textures.

Moreover, the authors [33] have updated the datasets of Rain100H and Rain100L, both of which include 1,800 training pairs and 200 testing images. We notate them as RainHeavy* and RainLight*, respectively. And we take more recent state-of-the-art methods into comparison on RainHeavy* and RainLight*, including DDN [24], SIRR [28], SPANet [26], RESCAN [5], PReNet [32] and JORDER-E [33]. Due to the inconsistent evaluation metrics adopted in their original papers, we re-train all these models for RainHeavy* and RainLight*. Average PSNR and SSIM values are reported in Table VII. One can see that our BRN achieves much higher quantitative

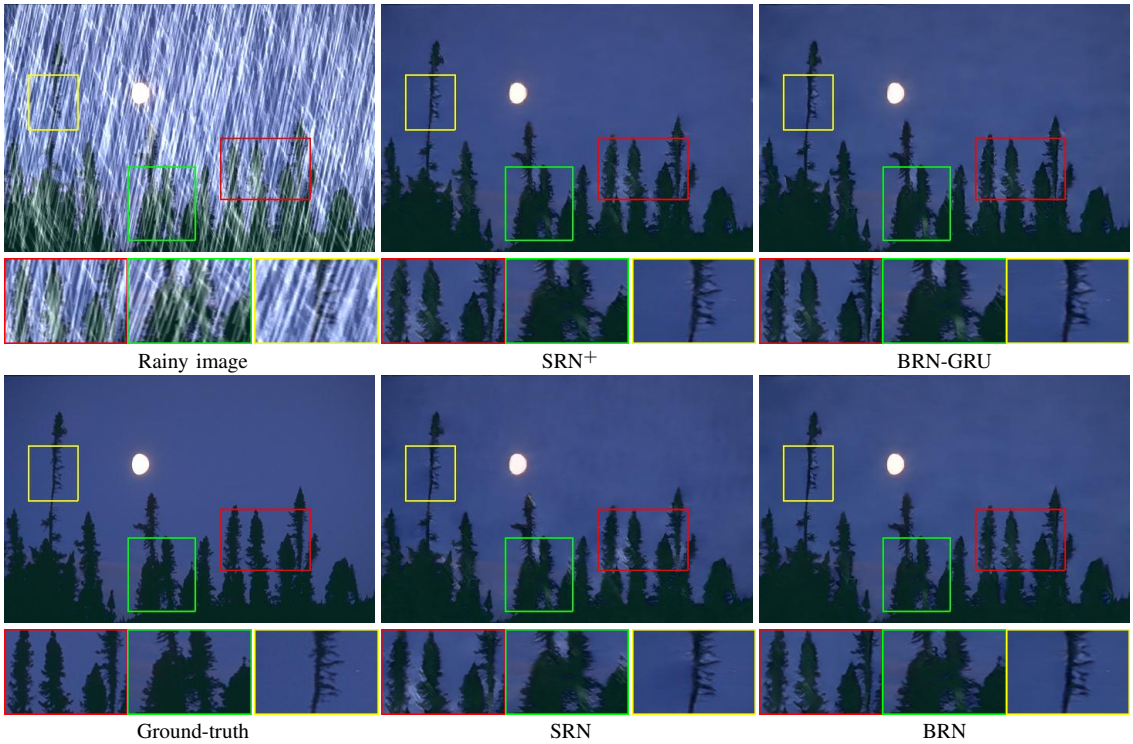

Fig. 7. Visual comparison of variants of SRN and BRN. SRN$^+$ is a variant of SRN by increasing ResBlocks number to 18. BRN-GRU is a variant of BRN by substituting LSTMs with GRUs.

TABLE V
AVERAGE PSNR AND SSIM COMPARISON ON SYNTHETIC DATASETS, INCLUDING RAIN100H [7], RAIN100L [7] AND RAIN12 [4].

	GMM [4]	DDN [24]	ResGuideNet [6]	JORDER [7]	RESCAN [5]	SIRR [28]	BRN
Rain100H	15.05/0.425	21.92/0.764	25.25/0.841	26.54/0.835	28.64/0.864	22.47/0.716	30.47/0.913
Rain100L	28.66/0.865	32.16/0.936	33.16/0.963	36.61/0.974	—	32.37/0.926	38.16/0.982
Rain12	32.02/0.855	31.78/0.900	29.45/0.938	33.92/0.953	—	34.02/0.935	36.74/0.959

Fig. 8. Comparison of extracted rain streak layers by BRN and RESCAN [5]. The rain streak layer by RESCAN contains background textures, while the one by BRN is clearer.

TABLE VII
AVERAGE PSNR AND SSIM COMPARISON ON NEW DATASETS OF RAIN100H AND RAIN100L, I.E., RAINHEAVY* AND RAINLIGHT* [33].

	DDN [24]	RESCAN [5]	SIRR [28]	SPANet [26]	PReNet [32]	JORDER-E [33]	BRN
RainHeavy*	22.03/0.713	28.02/0.862	22.17/0.719	26.59/0.869	29.36/0.903	29.21/0.891	30.27/0.917
RainLight*	31.66/0.922	38.43/0.982	32.20/0.929	36.13/0.975	37.93/0.983	39.13/0.985	38.86/0.985

TABLE VIII
GENERALIZATION EVALUATION ON REAL-WORLD RAINY DATASET SPADATA [26], WHERE THE MODELS ARE TRAINED FOR RAINLIGHT* [33].

	DDN [24]	RESCAN [5]	SIRR [28]	SPANet [26]	PReNet [32]	JORDER-E [33]	BRN
PSNR	34.80	34.73	34.84	35.26	35.00	34.13	35.34
SSIM	0.936	0.937	0.936	0.945	0.941	0.934	0.945

TABLE IX
COMPARISON OF RUNNING TIME (SECONDS).

Image Size	DDN [24]	RESCAN [5]	SIRR [28]	SPANet [26]	PReNet [32]	JORDER-E [33]	BRN
512×512	1.80	1.03	1.81	2.28	0.17	0.53	0.40
1024×1024	2.78	3.79	2.75	10.38	0.62	2.15	1.49

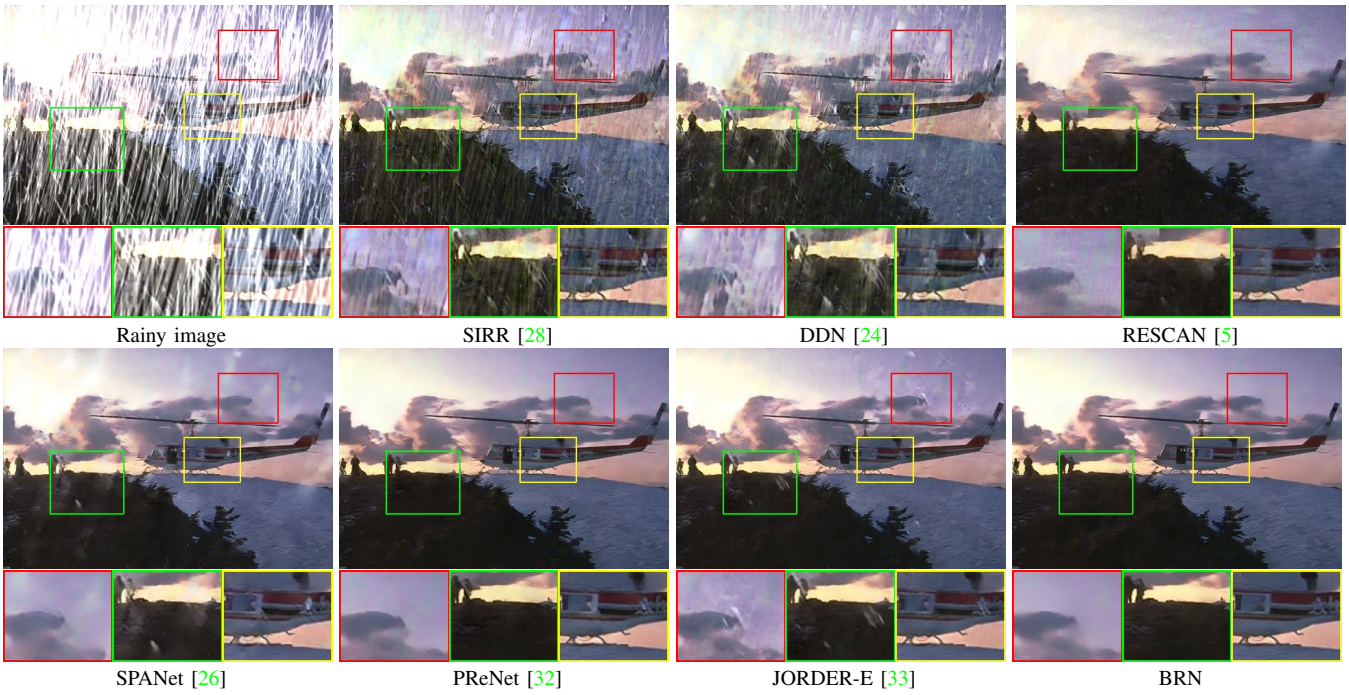

Fig. 9. Visual quality comparison of synthetic rainy image in RainHeavy* [33].

metrics than all the competing methods on RainHeavy*. On RainLight*, BRN is only a little inferior to JORDER-E in terms of PSNR, but is comparable in terms of SSIM. In Fig. 9, BRN can remove rain streaks clearly.

Then we evaluate BRN on another dataset Rain1400 [24], including 12,600 rainy images for training and 1,400 rainy images for testing. As reported in Table VI, BRN achieves more significant gains than DDN and SIRR. The visual quality improvement is also very significant, while the results by DDN and SIRR still have visible rain streaks in Fig. 10. The performance improvements in quantitative metrics and visual quality can be attributed to the bilateral modeling capacity of BRN.

Finally, the comparison of computational efficiency is conducted. Considering optimization-based method GMM is very time-consuming and the code of ResGuideNet is not released, we report the inference time of DDN [24], RESCAN [5], SIRR [28], SPANet [26], PReNet [32] and JORDER-E [33] for image sizes 512×512 and 1024×1024. The running time is recorded on one NVIDIA GTX 1080Ti GPU. As reported in Table IX, BRN is more efficient than the competing methods except our previous PReNet [32]. Meanwhile, considering better performance in Table VII and better generalization ability in Table VIII, our BRN can be a preferred choice for practical applications.

C. Evaluation on Real Rainy Images

In this subsection, we evaluate the generalization ability of BRN against state-of-the-art competing algorithms on real-world rainy images. The authors in [26] established a real-world rainy dataset, i.e., SPAData, containing 1,000 testing rainy and clean image pairs from rainy videos. Our methods are compared with DDN [24], RESCAN [5], SIRR [28], SPANet [26], PReNet [32] and JORDER-E [33]. We note that the semi-supervised SIRR [28] network is fine-tuned using many real-world rainy images in unsupervised learning, while the other supervised methods are trained for RainLight*. The quantitative metrics are reported in Table VIII, from which one can see that BRN can generalize well to real-world rainy images. From Tables VII and VIII, albeit BRN is slightly inferior to JORDER-E on RainLight*, it achieves much higher metrics (i.e., more than 1dB in PSNR) when applying to real rainy images. As shown in Fig. 11, BRN is able to clearly remove rain streaks, while the other methods suffer from remaining visible rain streaks.

V. CONCLUSION

In this paper, we first proposed a simple yet effective single recurrent network (i.e., SRN) for image deraining. SRN is effective not only in learning residual mapping for extracting rain streaks but also in learning direct mapping for predicting background image. Then by coupling two SRNs and

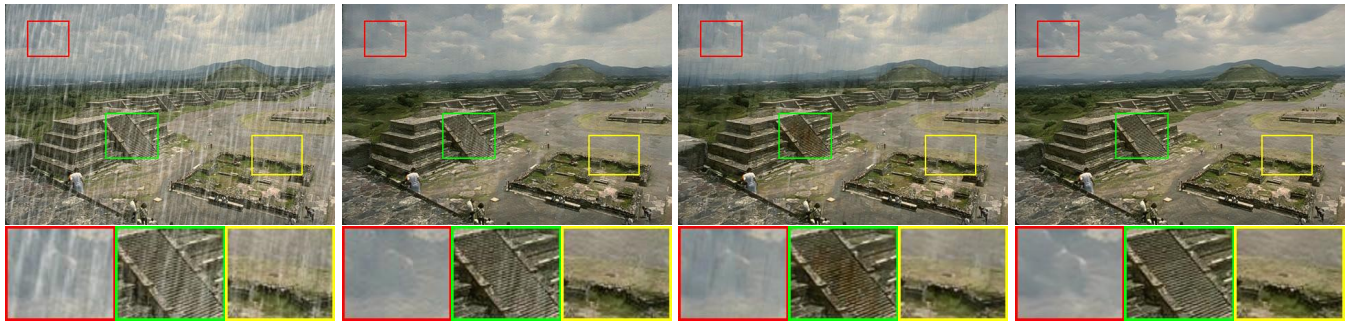

Fig. 10. Visual comparison on Rain1400 dataset [24].

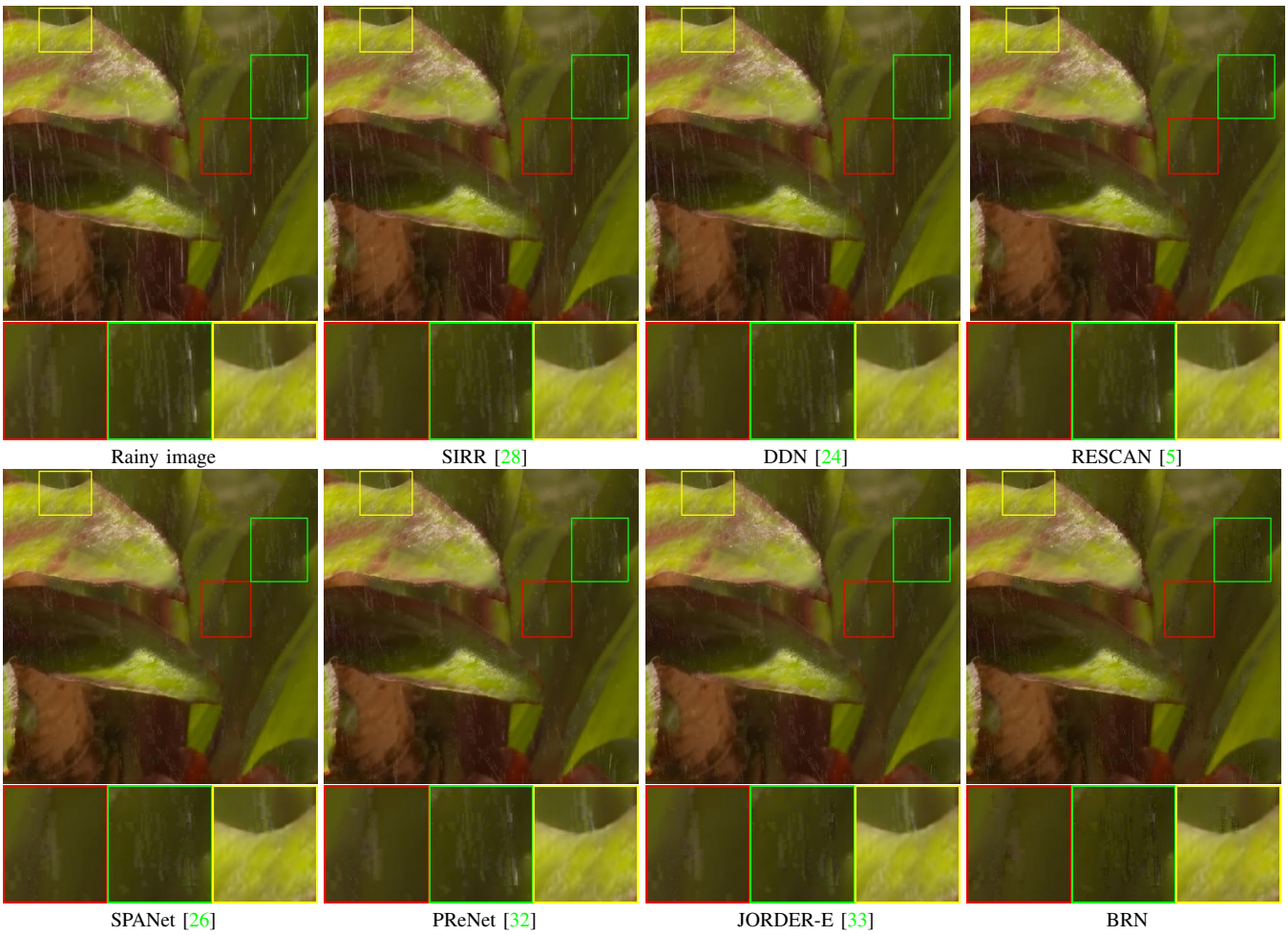

Fig. 11. Visual comparison of real-world rainy image in SPAData [26].

incorporating BLSTMs, BRN was proposed to sequentially extract rain streaks and generate clean background image, where the interplay between rain streak layer and clean background image layer can be exploited. Both SRN and BRN can be easily trained using only a single negative SSIM loss. Extensive experimental results validate the superiority of BRN in deraining performance on several benchmark datasets against state-of-the-art deraining methods. And our trained models can be better generalized to real-world rainy images. Additionally, the ideas of progressive and BLSTMs models may also be beneficial to other layer separation tasks, e.g., reflection removal, shadow removal and fences removal, which will be left for further studies in future work.

REFERENCES

[1] X. Fu, B. Liang, Y. Huang, X. Ding, and J. Paisley, "Lightweight pyramid networks for image deraining," *IEEE Transactions on Neural Networks and Learning Systems*, 2019. 1, 2, 3, 5

[2] 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*, vol. 21, no. 4, p. 1742, 2012. 1

[3] 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*, vol. 26, no. 6, pp. 2944–2956, 2017. 1, 3

[4] 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*, 2016, pp. 2736–2744. 1, 2, 7, 8

[5] 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*, 2018, pp. 262–277. 1, 2, 3, 6, 7, 8, 9, 10

[6] Z. Fan, H. Wu, X. Fu, Y. Hunag, and X. Ding, "Residual-guide feature fusion network for single image deraining," in *ACM Multimedia*, 2018. 1, 3, 5, 7, 8

[7] 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*, 2017, pp. 1357–1366. 1, 2, 3, 5, 6, 7, 8

[8] D. Ren, W. Zuo, D. Zhang, L. Zhang, and M.-H. Yang, "Simultaneous fidelity and regularization learning for image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 1

[9] H. Wang, M. Li, Y. Wu, Q. Zhao, and D. Meng, "A survey on rain removal from video and single image," *arXiv preprint arXiv:1909.08326*, 2019. 1

[10] 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 Computer Vision*, 2013, pp. 1968–1975. 1, 2

[11] Y. Luo, Y. Xu, and H. Ji, "Removing rain from a single image via discriminative sparse coding," in *Proceedings of the IEEE International Conference Computer Vision*, 2015, pp. 3397–3405. 1, 2

[12] 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 Computer Vision*, 2017, pp. 1717–1725. 1

[13] 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*, vol. 26, no. 7, pp. 3142–3155, 2017. 1

[14] W. Dong, P. Wang, W. Yin, G. Shi, F. Wu, and X. Lu, "Denoising prior driven deep neural network for image restoration," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 10, pp. 2305–2318, 2018. 1

[15] C. Tian, Y. Xu, Z. Li, W. Zuo, L. Fei, and H. Liu, "Attention-guided cnn for image denoising," *Neural Networks*, 2020. 1

[16] J. Xu, L. Zhang, and D. Zhang, "External prior guided internal prior learning for real-world noisy image denoising," *IEEE Transactions on Image Processing*, vol. 27, no. 6, pp. 2996–3010, 2018. 1

[17] C. Tian, Y. Xu, and W. Zuo, "Image denoising using deep cnn with batch renormalization," *Neural Networks*, vol. 121, pp. 461–473, 2020. 1

[18] 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*, vol. 38, no. 2, pp. 295–307, 2016. 1

[19] 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

[20] 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*, 2016, pp. 1637–1645. 1

[21] 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*, 2017. 1

[22] X. Tao, H. Gao, X. Shen, J. Wang, and J. Jia, "Scale-recurrent network for deep image deblurring," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8174–8182. 1

[23] J. Zhang, J. Pan, J. Ren, Y. Song, L. Bao, R. W. Lau, and M.-H. Yang, "Dynamic scene deblurring using spatially variant recurrent neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1

[24] 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*, 2017, pp. 1715–1723. 1, 2, 3, 7, 8, 9, 10

[25] H. Zhang and V. M. Patel, "Density-aware single image de-raining using a multi-stream dense network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 695–704. 1, 3

[26] T. Wang, X. Yang, K. Xu, S. Chen, Q. Zhang, and R. W. Lau, "Spatial attentive single-image deraining with a high quality real rain dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 270–12 279. 1, 2, 7, 8, 9, 10

[27] H. Zhu, X. Peng, J. T. Zhou, S. Yang, V. Chanderasekh, L. Li, and J.-H. Lim, "Singe image rain removal with unpaired information: A differentiable programming perspective," in *AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 9332–9339. 1, 3

[28] W. Wei, D. Meng, Q. Zhao, Z. Xu, and Y. Wu, "Semi-supervised transfer learning for image rain removal," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3877–3886. 2, 3, 7, 8, 9, 10

[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 2, 3

[30] 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*, 2015, pp. 802–810. 2, 3

[31] 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*, vol. 13, no. 4, pp. 600–612, 2004. 2, 5

[32] D. Ren, W. Zuo, Q. Hu, P. Zhu, and D. Meng, "Progressive image deraining networks: A better and simpler baseline," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2, 7, 8, 9, 10

[33] W. Yang, R. T. Tan, J. Feng, J. Liu, S. Yan, and Z. Guo, "Joint rain detection and removal from a single image with contextualized deep networks," *IEEE transactions on pattern analysis and machine intelligence*, 2019. 2, 7, 8, 9, 10

[34] 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 Computer Vision*, 2017, pp. 2526–2534. 2

[35] 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. 2

[36] 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 IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[37] ——, "Fastderain: A novel video rain streak removal method using directional gradient priors," *IEEE Transactions on Image Processing*, vol. 28, no. 4, pp. 2089–2102, 2018. 2

[38] 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*, vol. 24, no. 9, pp. 2658–2670, 2015. 2

[39] 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*, 2018, pp. 6644–6653. 2

[40] H. Zhang, V. Sindagi, and V. M. Patel, "Image de-raining using a conditional generative adversarial network," *IEEE Transactions on Circuits and Systems for Video Technology*, 2019. 3, 5

[41] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *International Conference on Machine Learning*, 2010, pp. 807–814. 3, 4

[42] 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," *Conference on Empirical Methods in Natural Language Processing*, 2014. 3, 7

[43] 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

[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015. 5