Joint Rain Detection and Removal from a Single Image with Contextualized Deep Networks

Wenhan Yang, *Member, IEEE*, Robby T. Tan, *Member, IEEE*, Jiashi Feng, *Member, IEEE*, Zongming Guo, *Member, IEEE*, Shuicheng Yan, *Fellow, IEEE*, and Jiaying Liu, *Senior Member, IEEE*

Abstract—Rain streaks, particularly in heavy rain, not only degrade visibility but also make many computer vision algorithms fail to function properly. In this paper, we address this visibility problem by focusing on single-image rain removal, even in the presence of dense rain streaks and rain-streak accumulation, which is visually similar to mist or fog. To achieve this, we introduce a new rain model and a deep learning architecture. Our rain model incorporates a binary rain map indicating rain-streak regions, and accommodates various shapes, directions, and sizes of overlapping rain streaks, as well as rain accumulation, to model heavy rain. Based on this model, we construct a multi-task deep network, which jointly learns three targets: the binary rain-streak map, rain streak layers, and clean background, which is our ultimate output. To generate features that can be invariant to rain steaks, we introduce a contextual dilated network, which is able to exploit regional contextual information. To handle various shapes and directions of overlapping rain streaks, our strategy is to utilize a recurrent process that progressively removes rain streaks. Our binary map provides a constraint and thus additional information to train our network. Extensive evaluation on real images, particularly in heavy rain, shows the effectiveness of our model and architecture.

Index Terms—Rain removal, rain detection, deep learning, rain accumulation, contextualized dilated network

1 Introduction

Restoring images degraded by rain is beneficial to many computer vision applications for outdoor scenes. Rain reduces visibility significantly, which can impair many computer vision systems. There are two types of visibility degradation brought by rains. Distant rain streaks accumulate and exhibit atmospheric veiling effects visually similar to mist or fog. The rain streaks accumulation scatters light out and into the line of sight, and severely reduce the visibility. Nearby rain streaks generate specular highlights and occlude background scenes. These rain streaks are diversified in shapes, sizes, and directions, particularly in heavy rain, introducing severe visibility degradation.

Many methods have been proposed to address the restoration of rain degradation. Some focus on video deraining [3], [4], [5], [6], [7], [8], [9], [10], [11]. Others focus on single image rain removal. These methods regard the rain streak removal problem as a signal separation problem [11], [12], [13], [14], [15], [16], or by relying on nonlocal mean smoothing [17]. These methods have made progress to some extent, however, they still suffer from some limitations. Because the rain streaks and background textures are overlapped intrinsically in the feature space, most methods cause non-rain regions to lose texture details and be over-smoothed.

The rain degradation can be complex, and previous widely used rain models (e.g., [11], [12]) neglect some important visual factors of real rain images, such as the atmospheric veils caused by rain streak accumulation, and different shapes or directions of streaks. Moreover, many existing algorithms operate in a patch-wise way with a limited receptive field (a limited spatial range). Thus, spatial contextual information in larger regions is absent, which in fact has been proven to be useful for rain removal [18].

To address these limitations, we make effort to develop a novel rain model that explicitly describes various rain conditions in real scenes, including rain streak accumulation and heavy rain, and then, design an effective deep learning architecture based on the novel rain model. Here, we focus on a single input image. Our ideas are as follows.

First, we present novel region-dependent rain models. A rain-streak binary map, where '1' indicates the presence of individually visible rain streaks, and '0' otherwise, is injected to model the location information of rain streaks. To simulate heavy rains, our rain model also considers the appearance of rain streak accumulation, and the various shapes and directions of overlapping streaks.

Second, with the new proposed rain model, a deep network is built to detects and removes rain jointly. The automatically detected rain streak regions provide useful information to constrain the rain removal, and enable the network to perform an

- *W. Yang, Z. Guo, and J. Liu are with the Institute of Computer Science and Technology, Peking University, Beijing 100871, China. E-mail: {yangwenhan, guozongming, liujiaying}@pku.edu.cn.*
- *R. T. Tan is with the Yale-NUS College and the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 119077. E-mail: tanrobby@gmail.com.*
- *J. Feng is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 119077. E-mail: elefjia@nus.edu.sg.*
- *S. Yan is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 119077, and also with the Qihoo/360 company, Beijing, China. E-mail: eleyans@nus.edu.sg.*

Manuscript received 11 Dec. 2017; revised 2 Nov. 2018; accepted 10 Jan. 2019. Date of publication 28 Jan. 2019; date of current version 5 May 2020. (Corresponding author: Jiaying Liu.)
Recommended for acceptance by P. Favaro.
Digital Object Identifier no. 10.1109/TPAMI.2019.2895793

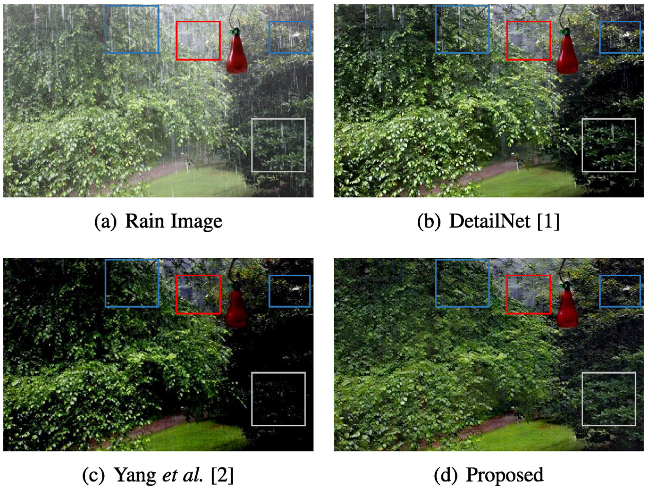

(a) Rain Image
(b) DetailNet [1]
(c) Yang *et al.* [2]
(d) Proposed

Fig. 1. Visual comparison of different methods to remove heavy rain streaks and enhance the visibility. Regions in blue boxes show our superiority in rain streak removal. Regions in red boxes show our superiority in rain accumulation removal. Regions in gray boxes show our superiority in detail preservations. Our method significantly outperforms DetailNet [1] and our previous work [2].

adaptive operation on rain and non-rain regions, preserving richer details.

Third, a contextualized dilated network is proposed to enlarge the receptive field and to get context information from a larger region. The features in this network are refined recurrently. In each recurrence, the output features are the aggregated from different convolution paths with different dilated factors.

Finally, to restore images captured in heavy rain cases with both rain accumulation and various rain streak directions, a recurrent rain detection and removal network is constructed to progressively removes rain streaks. Extensive experiments are conducted to demonstrate the superiority of our method on both synthesized data and real data. Particularly for some heavy rain images, our method achieves considerably good results.

Hence, our contributions are:

1) The first method to inject rain location information into the rain model, and also to model the atmospheric veils caused by rain streak accumulation as well as various shapes and directions of overlapping rain streaks. The new model provides more visually realistic rain data for training.
2) The first method to jointly detect and remove rains from single images. With detected rain location information, our method provides better rain removal results.
3) The first rain removal method that takes a contextualized dilated network as its backbone to obtain more context and reconstruct rich local details.
4) The first method that aims to handle heavy rain using a recurrent rain detection and removal network, obtaining good results even in significantly complex cases.

Note that, this paper is the extension of our earlier publication [2]. We summarize the changes Here. First, in our detail preserving rain accumulation removal method (Section 5.2), we not only deal with the veiling effect but also restore image brightness, which further enhances the visibility of the rain removal results as shown in Fig. 1. Second, in our recurrent joint rain detection and removal, we employ residual task learning (Section 5.1). In the second recurrence, we re-estimate the rain streak and background image based on the input rain image and the estimation in the last recurrence. The estimated variables and features are forwarded to the current recurrence to force the current sub-network to learn the residual features and variables, which leads to a significant gain. Fourth, to properly train our joint rain detection and removal network, we propose a coarse-to-fine multiscale loss (Section 4.3), which regularizes a subset of our contextualized dilated network to perform rain streak removal. Besides, we replace the commonly used MSE loss with the L_1 loss (Section 4.3), which also provides a small performance improvement. This leads to an improved performance. Moreover, we provide the details of our implementation, as well as more comprehensive analysis and evaluation.

2 Related Work

Rain image recovery [3], [4], [5], [6], [7], [8], [9], [10], [11] from video sequences has been widely explored. Garg et al. [3], [4], [5], [6], [19] first construct the appearance model to describe rain streaks and exploit it to detect rain pixels in video. Zhang et al. [7] and Brewer et al. [8] focus on the chromaticity and shape of rain streaks, respectively. Other methods construct novel features to model and detect rain streaks, such as frequency domain analysis [9], histogram of orientation [10] and generalized low rank [11]. These methods make full use of the rich information in videos and the temporal redundancy in adjacent frames to identify rain streaks. In contrast, our method attempts to jointly detect and remove rain regions from only a single image.

Compared with the video based deraining problem, the single image based problem is more ill-posed, due to the lack of temporal information. Some single-image based rain removal methods regard the problem as a layer separation problem. Huang et al. [12] attempt to separate the rain streaks from the high frequency layer by sparse coding, with a learned dictionary from the HOG features. However, the capacity of the morphological component analysis, the layer separation, and learned dictionary are limited. Thus, it usually causes over-smoothness of the background. In [11], a generalized low rank model is proposed, where the rain streak layer is assumed to be low rank. Kim et al. [20] first detect rain streaks and then remove them with the nonlocal mean filter. Luo et al. [15] propose a discriminative sparse coding method to separate rain streaks from background images. In [21], Li et al. exploits the Gaussian mixture models to separate the rain streaks, achieving the state-of-the-art performance, however, still with slightly smooth background. In [1], [22], a deep network that takes the image detail layer as its input and predicts the negative residues is constructed. It has a good capacity to keep texture details. However, it cannot handle the heavy rain cases when rain streaks are dense and significant. In [16], a novel convolutional neural network based on wavelet and dark channel is proposed to jointly remove rain streaks and haze. In this paper, we use the deep network to perform joint rain detection and removal, with the priors and constraints learned automatically from the synthesized data, and aim to address the issue of heavy rain removal.

In recent years, deep learning-based image processing applications emerged with promising performance. These applications include denoising [23], image completion [24], super-resolution [25], [26], [27], [28], [29], [30], deblurring [31], deconvolution [32], style transfer [33], compression artifacts removal [34], [35], etc. There are also some recent works on bad weather restoration or image enhancement, such as dehazing [36], [37], raindrop and dirt removal [38], [39], light enhancement [40], [41] and moderate rain removal [1], [22], [42], [43]. With the superior modeling capacity than shallow models, deep-learning based methods begin to solve harder problems, such as blind image denoising [23], image quality assessment [44], [45], image compression [46], and video coding [47], [48], [49], [50]. In this paper, we use deep learning to jointly detect and remove rain.

3 Region-Dependent Rain Image Model

The widely used rain model [15], [18], [21] is expressed as:

$$\mathbf{O} = \mathbf{B} + \widetilde{\mathbf{S}}, \tag{1}$$

where $\mathbf{B}$ is the background layer, and $\widetilde{\mathbf{S}}$ is the rain streak layer. $\mathbf{O}$ is the input image with rain streaks. Eq. (1) suffers from two deficiencies in rain modeling. First, $\widetilde{\mathbf{S}}$ can have a heterogeneous density, which is hard to be modeled by a uniform distribution. Second, mixed modeling rain and non-rain regions may lead to over-smoothness on the non-rain regions.

To overcome these drawbacks, we propose a generalized rain model that depicts rain streak location and rain intensity separately to fill the blank of previous works [1], [21], [51] as follows,

$$\mathbf{O} = \mathbf{B} + \mathbf{S} \circ \mathbf{R}, \tag{2}$$

which includes a new region-dependent variable $\mathbf{R}$ to indicate the locations of individually visible rain streaks, where $\circ$ means element-wise multiplication. Here, elements in $\mathbf{R}$ are binary values, where '1' indicates rain regions and '0' indicates non-rain regions. The new model provides two benefits: (1) it gives additional information for the network to learn about rain streak regions, (2) it allows a new rain removal pipeline to detect rain regions first, and then to operate differently on rain-streak and non-rain-streak regions, preserving background details.

In the real world, rain appearance is not only formed by individual rain streaks, but also by accumulation of rain streaks. When rain accumulation is dense, the individual streaks cannot be observed clearly. Aside from rain accumulation, in many occasions, particularly in heavy rain, rain streaks can have various shapes and directions that overlap to each other.

To accommodate these two phenomena (i.e., rain streak accumulation and overlapping rain streaks with different directions), we create a new model. The model comprises of multiple layers of rain streaks, representing the diversity of rain streaks. It also includes the appearance of rain accumulation, by relying on the Koschmieder model that is approximately applicable to many turbid media, including mist, fog (e.g., [52]) and underwater (e.g., [53], [54]). Our new rain model is expressed as:

$$\mathbf{O} = \alpha\left(\mathbf{B} + \sum_{t_r=1}^{s} \widetilde{\mathbf{S}}_{t_r} \circ \mathbf{R}\right) + (1-\alpha)\mathbf{A}, \tag{3}$$

where each $\widetilde{\mathbf{S}}_{t_r}$ is a layer of rain streaks that have the same direction. t_r is the index of the rain-streak layers, and s is the maximum number of rain-streak layers. $\mathbf{A}$ is the global atmospheric light, α is the atmospheric transmission. Based on Eq. (3), we can generate synthetic images that are better representative of natural images than those generated by Eq. (1). Thus, we can use these images to train our network. Note that, the rain accumulation appearance is enforced on the rain-contaminated image ($\mathbf{B} + \sum_{t_r=1}^{s} \widetilde{\mathbf{S}}_{t_r} \circ \mathbf{R}$), hence Eq. (3) implies that, we can handle rain accumulation and rain streak removal separately, which provides convenience for our training.

4 Joint Rain Streak Detection and Removal

We construct a multi-task network to perform **JO**int **R**ain **DE**tection and **R**emoval (JORDER) that solves the inverse problem in Eq. (2) through end-to-end learning. Rain regions are first detected by JORDER to further constrain the rain removal. To leverage more context without losing local details, we propose a novel network structure – the contextualized dilated network – for extracting the rain discriminative features and facilitating the following rain detection and removal.

4.1 Multi-Task Networks for Joint Rain Detection and Removal

Relying on Eq. (2), given the observed rain image $\mathbf{O}$, our goal is to estimate $\mathbf{B}$, $\mathbf{S}$ and $\mathbf{R}$. Due to the ill-posedness nature of the problem, we employ a maximum-a-posteriori (MAP) estimation:

$$\arg\min_{\mathbf{B},\mathbf{S},\mathbf{R}} ||\mathbf{O} - \mathbf{B} - \mathbf{S} \circ \mathbf{R}||_2^2 + P_b(\mathbf{B}) + P_s(\mathbf{S}) + P_r(\mathbf{R}), \tag{4}$$

where $P_b(\mathbf{B})$, $P_s(\mathbf{S})$ and $P_r(\mathbf{R})$ are the enforced priors on $\mathbf{B}, \mathbf{S}$ and $\mathbf{R}$, respectively. Previous priors on $\mathbf{B}$ and $\mathbf{S}$ include hand-crafted features, e.g., cartoon texture decomposition [12], and some data-driven models, such as sparse dictionary [15] and Gaussian mixture models [21]. For deep learning methods, the priors of $\mathbf{B}, \mathbf{S}$ and $\mathbf{R}$ are learned from the training data and are embedded into the network implicitly.

The estimation of $\mathbf{B}, \mathbf{S}$ and $\mathbf{R}$ is intrinsically correlated. Thus, the estimation of $\mathbf{B}$ benefits from the predicted $\widehat{\mathbf{S}}$ and $\widehat{\mathbf{R}}$. To convey this, the natural choice is to employ a multi-task architecture, which can be trained using multiple loss functions based on the ground truths of $\mathbf{R}$, $\mathbf{S}$ and $\mathbf{B}$ (see the blue dash box in Fig. 2).

As shown in the figure, we first exploit a contextualized dilated network to extract the rain feature representation $\mathbf{F}$. Subsequently, $\mathbf{R}$, $\mathbf{S}$ and $\mathbf{B}$ are predicted in a sequential order, implying a continuous process of rain streak detection, estimation and removal. The input features for each task are the concatenation of the general feature $\mathbf{F}$, and the intermediate estimation results of the previous tasks. There are several potential choices for the network structures, such as estimating the three variables in the order of $\mathbf{S}$, $\mathbf{R}$, $\mathbf{B}$, or in parallel (instead of sequential). The effectiveness of these choices are evaluated in Section 6.

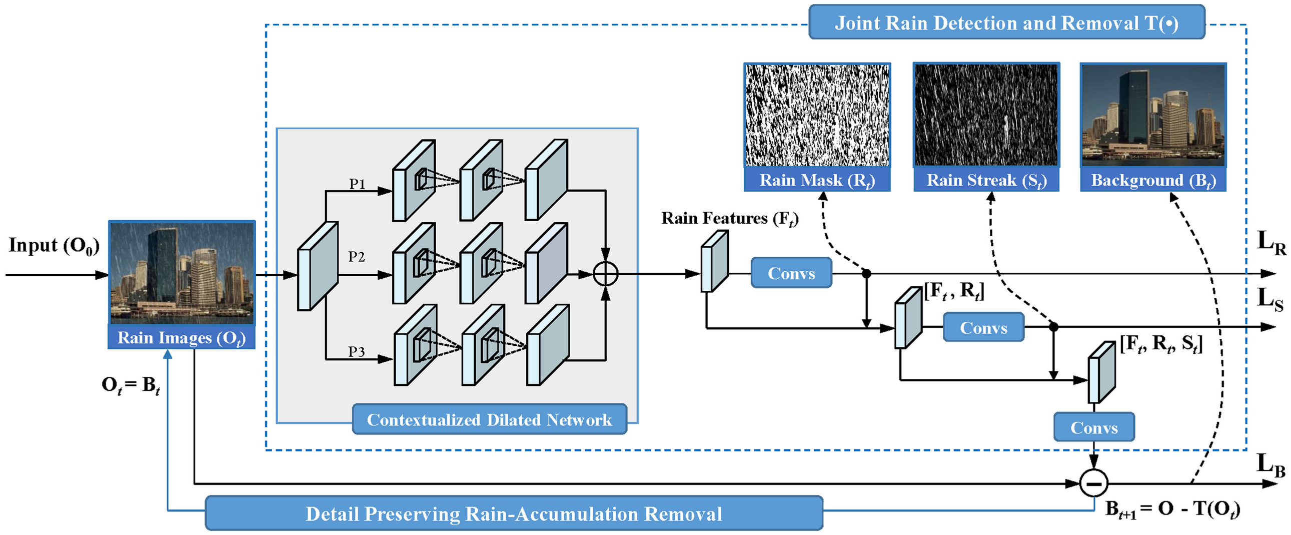


Fig. 2. The architecture of our rain removal method, including the proposed recurrent joint rain detection and removal, and the detail preserving rain-accumulation removal method placed between the two recurrences. Each recurrence is a multi-task network to perform a joint rain detection and removal (in the blue dash box). In such a network, a contextualized dilated network (in the gray region) extracts rain features $\mathbf{F}_t$ from the input rain image $\mathbf{O}_t$. Then, $\mathbf{R}_t$, $\mathbf{S}_t$ and $\mathbf{B}_t$ are predicted to perform joint rain detection, estimation and removal. Between the two recurrences, the detail preserving rain-accumulation removal is utilized to enhance the visibility. The features and estimated variables in the last recurrence are forward to the current one, and the sub-network in this recurrence stage only learns the residuals.

4.2 Contextualized Dilated Networks

For rain removal task, contextual information from an input image is demonstrated to be useful for automatically identifying and removing the rain streaks [18]. Thus, we propose a contextualized dilated network to aggregate context information at multiple scales for learning the rain features. The network gains contextual information in two ways: 1) through a recurrent structure, which is similar to the recurrent ResNet [55], and provides an increasingly larger receptive field for the subsequent layers; 2) in each recurrence, the output features aggregate the representations of the three convolution paths with different dilated factors and receptive fields.

Specifically, as shown in Fig. 3a, the network first transforms the input rain image into feature space via the first convolution. Then, the network refines the features progressively. In each recurrence, the results from the three convolution paths with different dilated factors are aggregated with the input features from the last recurrence via the identity forwarding. The dilated convolution [56] weights pixels with a step size of the dilated factor, and thus increases its receptive field without losing resolution. Our three dilated paths consist of two convolutions with the same kernel size 3×3. However, with different dilated factors, different paths have their own receptive field. As shown in the top part of the gray region in Fig. 2, path P_2 consists of two convolutions with the dilated factor 2. The convolution kernel is shown as the case of DF$= 2$. Thus, cascading two convolutions, the three paths have their receptive fields of $5 \times 5, 9 \times 9$ and 13×13.

To provide a formal description, let $\mathbf{f}^k_{\text{in}}$ denote the input feature map for the recurrent subnetwork at the kth time step. The output feature map $\mathbf{f}^k_{\text{out}}$ of the recurrent subnetwork is progressively updated as follows:

$$\mathbf{f}^k_{\text{out}} = \max\left(0, \sum_{t_p=1}^{3}\left(\mathbf{W}^k_{\text{mid},t_p} * \mathbf{f}^k_{\text{mid},t_p} + \mathbf{b}^k_{\text{mid},t_p}\right)\right) + \mathbf{f}^k_{\text{in}},$$

where $\mathbf{f}^k_{\text{mid},t_p} = \max\left(0, \mathbf{W}^k_{\text{in}} * \mathbf{f}^k_{\text{in},t_p} + \mathbf{b}^k_{\text{in},t_p}\right)$. $\mathbf{f}^k_{\text{in}} = \mathbf{f}^{k-1}_{\text{out}}$ is the output features by the recurrent subnetwork at $(k-1)$th time step. $*$ denotes the convolution operator. The iteration variable t_p denotes the sequence number of dilated convolution paths. Note that, the by-pass connection lies between $\mathbf{f}^k_{\text{in}}$ and $\mathbf{f}^k_{\text{out}}$. The feature map $\mathbf{f}^k_{\text{out}}$ can be viewed as the recovered kth layer details of the feature maps. Let K denote the total recurrence number of the sub-networks, then the relation between $\mathbf{f}^1_{\text{in}}$, $\mathbf{f}^K_{\text{out}}$ and the overall network is

$$\begin{aligned} \mathbf{f}^1_{\text{in}} &= \max(0, \mathbf{W}_{\text{input}} * \mathbf{f}_{\text{input}} + \mathbf{b}_{\text{input}}), \\ \mathbf{F} &= \mathbf{f}^K_{\text{out}}, \end{aligned} \tag{5}$$

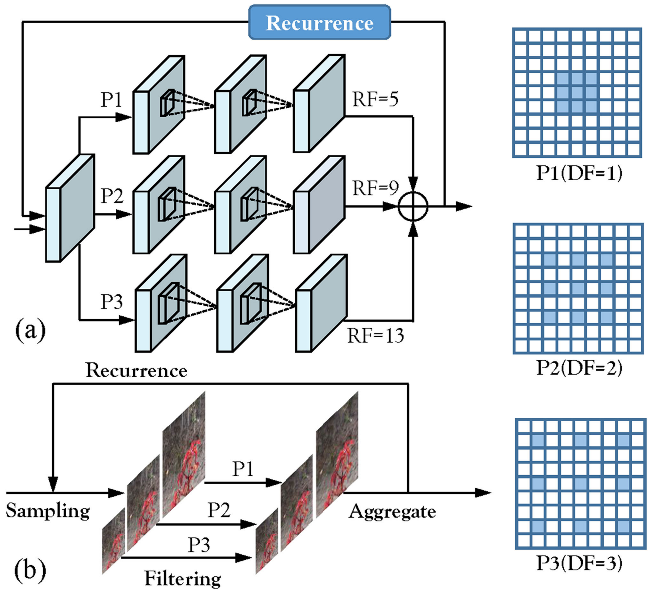


Fig. 3. The architecture of contextualized dilated networks. (a) The contextualized detailed network structure and illustration for the receptive fields (RF). (b) The corresponding conceptual explanation for the functionality of contextualized dilated networks.

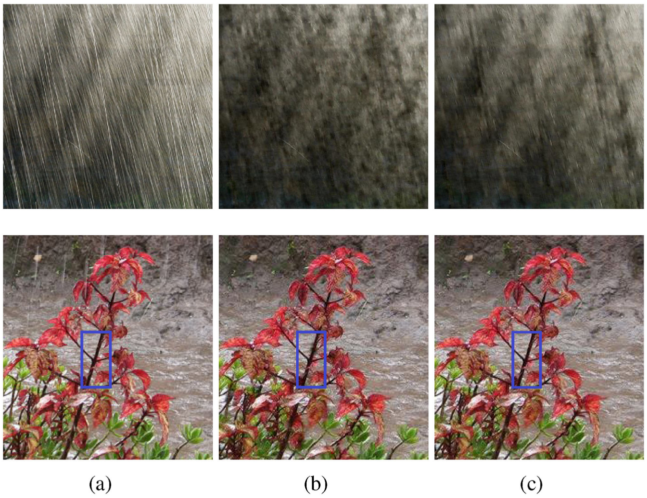


Fig. 4. Comparison of rain removal results with and without the contextualized dilation. (a) Rain images. (b) The results generated by the network without the contextualized dilation. (c) The results generated by the network with the contextualized dilation.

where $\mathbf{W}_{\mathrm{input}}$ and $\mathbf{b}_{\mathrm{input}}$ denote the filter parameter and basis of the convolution layer before the recurrent subnetwork. Hence, $\mathbf{F}$ is the output general features of the contextualized dilated networks, as one of the inputs of successive subnetworks.

Besides enlarging the receptive fields, the architecture in Fig. 3a is also meaningful from the perspective of traditional signal processing. As shown in Fig. 3b, it behaves similar to hierarchal multi-scale filters across scales. The signal is first split into different scales. They are, then, filtered separately and aggregated. This process is repeated throughout the network. As shown in Fig. 4, with the larger receptive fields and using the multi-scale cues, our contextualized dilations help construct more locally consistent results with less holes and are capable of preserving local details, as shown in the 'stem' regions (bounded by the blue box) in the bottom panel of the figure.

4.3 Network Training

Let $\mathbf{F}_{\mathrm{rr}}(\cdot), \mathbf{F}_{\mathrm{rs}}(\cdot)$ and $\mathbf{F}_{\mathrm{bg}}(\cdot)$ denote the inverse recovery functions modelled by the learned network to generate the estimated rain streak binary map $\widehat{\mathbf{R}}$, rain streak map $\widehat{\mathbf{S}}$ and background image $\widehat{\mathbf{B}}$ based on the input rain image $\mathbf{O}$. We use $\boldsymbol{\Theta}$ to collectively represent all the parameters of the network.

We use n sets of corresponding rain images, background images, rain region maps and rain streak maps $\left\{(\mathbf{o}_i, \mathbf{g}_i, \mathbf{r}_i, \mathbf{s}_i)\right\}_{i=1}^{n}$ for training. We adopt the following joint loss function to train the network parametrized by $\boldsymbol{\Theta}$ such that it is capable to jointly estimate $\mathbf{r}_i$, $\mathbf{s}_i$ and $\mathbf{g}_i$ based on rain image $\mathbf{o}_i$:

$$\begin{aligned} L(\boldsymbol{\Theta}) = \frac{1}{n}\sum_{i=1}^{n}\Big(&||\mathbf{F}_{\mathrm{rs}}(\mathbf{o}_i;\boldsymbol{\Theta}) - \mathbf{s}_i|| + \lambda_1||\mathbf{F}_{\mathrm{bg}}(\mathbf{o}_i;\boldsymbol{\Theta}) - \mathbf{g}_i|| \\ &-\lambda_2\big(\log \widehat{\mathbf{r}}_{i,1}\mathbf{r}_{i,1} + \log\left(1-\widehat{\mathbf{r}}_{i,2}\right)(1-\mathbf{r}_{i,2})\big)\Big), \end{aligned} \tag{6}$$

where:

$$\widehat{\mathbf{r}}_{i,j} = \frac{\exp\left\{\mathbf{F}_{\mathrm{rr},j}(\mathbf{o}_i;\boldsymbol{\Theta})\right\}}{\sum_{k=1}^{2}\exp\left\{\mathbf{F}_{\mathrm{rr},k}(\mathbf{o}_i;\boldsymbol{\Theta})\right\}}, j \in \{1,2\}.$$

Parameters λ_1 and λ_2 are the weighting factors. The network is trained to minimize the above loss, using the back-propagation. To better regularize the training of the contextualized dilated network with the recurrent multi-path structure, we further extend the loss function to be the combination of losses of several sub-networks. We use $\boldsymbol{\Theta}^1$ and $\boldsymbol{\Theta}^2$ to denote the parameters of the sub-networks with only path P1 and paths P1, P2 in each recurrence of Fig. 3, respectively. Then, the losses of these two sub-networks are denoted to $L^1(\boldsymbol{\Theta}^1)$ and $L^2(\boldsymbol{\Theta}^2)$. We train these two losses with $L(\boldsymbol{\Theta})$ together:

$$L^{\mathrm{a}}(\boldsymbol{\Theta}) = L(\boldsymbol{\Theta}) + L^1(\boldsymbol{\Theta}^1) + L^2(\boldsymbol{\Theta}^2). \tag{7}$$

5 Rain Removal in Real Image

In this section, we further enhance our network to handle both multiple rain-streak layers (where each layer has its own streak direction) and rain accumulation. Several JORDER networks are cascaded to perform progressive rain detection and removal and recover the background layer with increasingly better visibility.

5.1 Recurrent JORDER with Residue Task Learning

We define the process of the network $\mathbf{T}(\cdot)$ in the blue dash box of Fig. 2, which generates the rain streak based on the information of previous estimation $\mathbf{O}_t$ and the rain input $\mathbf{O}$. Then, our recurrent rain detection and removal works as follows,

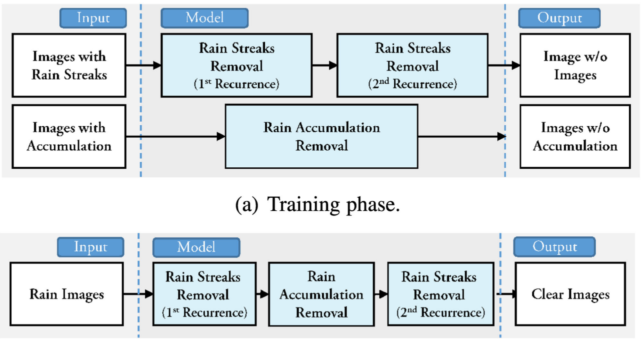
$$\begin{aligned} [\mathbf{r}_t, \Delta\mathbf{R}_t, \Delta\mathbf{S}_t, \Delta\mathbf{F}_t] &= \mathbf{T}(\mathbf{O}_t, \mathbf{O}), \\ \mathbf{R}_t &= \Delta\mathbf{R}_t + \mathbf{R}_{t-1}, \\ \mathbf{S}_t &= \Delta\mathbf{S}_t + \mathbf{S}_{t-1}, \\ \mathbf{F}_t &= \Delta\mathbf{F}_t + \mathbf{F}_{t-1}, \\ \mathbf{B}_t &= \mathbf{O} - \mathbf{r}_t, \\ \mathbf{O}_{t+1} &= \mathbf{B}_t, \end{aligned} \tag{8}$$

where $\mathbf{R}_0 = 0$, $\mathbf{S}_0 = 0$ and $\mathbf{F}_0 = 0$. In each iteration t, based on the rain input $\mathbf{O}$ and $\mathbf{O}_t$, the rain streak $\mathbf{r}_t$ is re-estimated. We find that, once the residual task learning is used, our original scheme which accumulates the predicted residual and propagates it to the final estimation via updating $\mathbf{O}_t$ and $\mathbf{B}_t$ becomes no more effective. We also forward the features of each residual block in $\mathbf{T}(\cdot)$ to the next one, enforcing $\mathbf{T}(\cdot)$ at the next stage only learns the residual features. In this way, the method removes rain streaks progressively, part by part, based on the intermediate results from the previous step. The complexity of rain removal in each iteration is consequently reduced, enabling better estimation, especially in the case of heavy rain.

5.2 Detail Preserving Rain-Accumulation Removal

Distant rain streaks accumulate and form rain atmospheric veil, which is visually similar to fog. It causes visibility degradation, and thus needs to be removed. We call this process rain-accumulation removal. Since the degradation effect and the model (Eq. (3)) are similar to those of fog, our rain-accumulation removal is essentially similar to defogging (e.g., [36]). Like in defogging, the output of our rain removal clears up the veiling effect and boosts the contrast.

Eq. (3) suggests that the rain-accumulation removal should be the first step in the whole process of deraining, simply as pre-processing. However, in real cases, it is more


Input
Model
Output
Images with Rain Streaks
Rain Streaks Removal (1st Recurrence)
Rain Streaks Removal (2nd Recurrence)
Image w/o Images
Images with Accumulation
Rain Accumulation Removal
Images w/o Accumulation
Input
Model
Output
Rain Images
Rain Streaks Removal (1st Recurrence)
Rain Accumulation Removal
Rain Streaks Removal (2nd Recurrence)
Clear Images

(a) Training phase.

(b) Testing phase.

Fig. 5. The training and testing paradigm of JORDER-R-DEVEIL.

complicated. Since, when we apply the rain-accumulation removal as pre-processing, it degrades the quality of rain streaks. All rain streaks, including those that are already sharp and clearly visible, are further boosted, causing the streaks to look different from those in the training images. To address this problem, instead of applying the rain-accumulation removal as pre-processing, we apply it progressively in combination with our rain-streak removal. First, we apply rain-streak removal, followed by rain-accumulation removal, and then apply rain-streak removal again. This, as it turns out, is beneficial, since the rain-accumulation removal will make the appearance of less obvious rain streaks (which are likely unnoticed by the first round of the streak removal) become more apparent.

In a rainy day, particularly in heavy rain, the scenes are darker than those in a clear day. This is because usually the sky is overcast. Yet, the veiling effect (or the backward scattering) of rain accumulation increases the brightness of the scene. Because of this, when we remove the veiling effect, the outputs of our rain-accumulation removal can be perceived as darker than normal. To overcome this problem: First, in generating synthetic training data, we decrease the brightness of our images before adding the veiling effect. With this, we want to tell the network about the darkening effect due to the overcast sky. Second, in every epoch of training, we additionally add two purely white and black training pairs into the training set of the current epoch. This operation makes the network capable of generating results with white and black colors. It equals to enforcing color consistency constraint [57], [58], [59], [60] on the networks, which is beneficial for decreasing the color bias of the training data and generating naturally looking results in colors. For our rain-accumulation removal, we create another network based on contextualized dilated network (see the first inset in Fig. 2).

5.3 Network Training

We train the networks for rain streak and rain accumulation removal separately, but implement them jointly in the testing phase, as shown in Fig. 5. The reason of not using an end-to-end training is that the degradation of rain accumulation is multiplicative and its recovery usually has larger errors than those of rain streak removal. Thus, joint training can contaminate the recovery of rain streak removal, which in fact has a larger impact on human visual perception.

Our recurrent JORDAR network introduces an extra time variable t to the loss function $L^a(\mathbf{\Theta})$ in Eq. (7) and gives $L^a((\mathbf{\Theta}_t, t)$, where $L^a(\mathbf{\Theta}_0, 0) = L^a(\mathbf{\Theta}_0)$. When $t > 1$, $L^a(\mathbf{\Theta}_t, t)$ is equivalent to $L^a(\mathbf{\Theta})$ that replaces $\mathbf{o}_i$ and $\mathbf{\Theta}$ by $\mathbf{o}_{i,t}$ and $\mathbf{\Theta}_t$, respectively, where $\mathbf{o}_{i,t}$ is generated from the tth iterations of the process Eq. (8) on the initial $\mathbf{o}_i$. Then, the total loss function L^a_{Iter} for training $\mathbf{T}$ is

$$L^a_{\text{Iter}}(\{\mathbf{\Theta}_0, \ldots, \mathbf{\Theta}_\tau\}) = \sum_{t=0}^{\tau} L^a(\mathbf{\Theta}_t, t). \tag{9}$$

The rain-accumulation removal network is trained with the synthesized data generated by random transmission values and sampled background images. We calculate the transmission values based on depth information using, e.g., *Make3D*, an outdoor scene and 3D dataset [61].

Considering Eq. (3) and given the input $\mathbf{O}_{acc}$, the network estimates the transmission map $\hat{\alpha}$. Then, using $\mathbf{\Theta}_{acc}$ to collect the related parameters of the rain-accumulation removal network, the loss function is derived from:

$$L_{acc}(\Theta_{acc}) = \left\| \frac{\Theta_{acc} + \hat{\alpha} - 1}{\hat{\alpha} + \epsilon} - \mathbf{B}_{acc} \right\|_2^2, \tag{10}$$

where $\mathbf{B}_{acc}$ is the ground-truth accumulation-free image, ϵ is a positive small float number, set as 0.00001. Following the synthesis configuration of DehazeNet [36], the atmospheric light, $\mathbf{A}$, is set as a matrix full of one. This setting is the hardest case. Training with it makes the network not only capable of handling hard cases, but also achieve greater robustness to other settings in practice. Note that, in testing phase, $\mathbf{A}$ is estimated online.

6 Experimental Results

Datasets. We compare our method with state-of-the-art methods on a few benchmark datasets: (1) *Rain12*[1] [21], which includes 12 synthesized rain images with only one type of rain streaks; *Rain100L*, which is the synthesized data set with only one type of rain streaks; (2) *Rain20L*, which is a subset of Rain100L; (3) *Rain100H*, which is our synthesized data set with five streak directions. Note, while it is rare for a real rain image to contain rain streaks in many different directions, synthesizing this kind of images for training can boost the capacity of the network.

The images for synthesizing *Rain100L*, *Rain20L* and *Rain100H* are selected from BSD200 [62]. The dataset for training our network and another deep learning baseline – SRCNN for deraining – is BSD300, excluding the ones appeared in *Rain12*. The rain streaks are synthesized in two ways: (1) the photorealistic rendering techniques proposed by [5]; (2) the simulated sharp line streaks along a certain direction with a small variation within an image. The testing rain images are taken from the previous publications [15], [21], and selected from Google, Bing and Baidu search engines. The testing images show dense rain streaks and most of them also show rain accumulation. We release our training and testing sets, as well as our image rendering code to public.

Baseline Methods. We compare the four versions of our approaches, JORDER- (one version that has only one convolution path in each recurrence without using dilated convolutions), JORDER (Section 4 of [2]), JORDER-R (Section 5.1 of [2]),

[1] http://yu-li.github.io/

TABLE 1
PSNR Results Among Different Methods

Baseline	*Rain12*	*Rain100L*	*Rain100H*	*Rain800*
ID [12]	27.21	23.13	13.78	20.54
DSC [15]	30.02	24.16	15.66	22.46
LP [21]	32.02	29.11	14.26	23.68
CNN [38]	26.65	23.70	13.21	23.95
SRCNN [25]	34.41	32.63	18.29	25.10
DetailNet [1]	35.31	33.50	20.12	25.22
UGSM [64]	33.30	28.83	13.40	23.12
JCAS [65]	33.09	29.91	14.26	22.25
DID-MDN [66]	30.14	28.27	13.85	22.55
ID-CGAN [67]	20.78	23.39	16.86	23.81
JORDER-	35.86	35.41	20.79	25.61
JORDER	36.02	36.11	22.15	26.03
JORDER-R	**36.21**	36.62	23.45	26.73
JORDER-E	36.14	**37.10**	**24.54**	**27.08**

TABLE 2
SSIM Results Among Different Methods

Baseline	*Rain12*	*Rain100L*	*Rain100H*	*Rain800*
ID [12]	0.7534	0.6991	0.3968	0.6739
DSC [15]	0.8679	0.8663	0.5444	0.7060
LP [21]	0.9082	0.8812	0.4225	0.7954
CNN [38]	0.7829	0.8142	0.3712	0.6589
SRCNN [25]	0.9421	0.9392	0.6124	0.8232
DetailNet [1]	0.9485	0.9444	0.6351	0.8228
UGSM [64]	0.9323	0.8823	0.5089	0.7675
JCAS [65]	0.9276	0.9041	0.4837	0.7682
DID-MDN [66]	0.8762	0.8569	0.3748	0.7639
ID-CGAN [67]	0.8519	0.8186	0.4921	0.8072
JORDER-	0.9534	0.9632	0.5978	0.8378
JORDER	0.9612	0.9741	0.6736	0.8501
JORDER-R	**0.9644**	**0.9820**	0.7490	0.8683
JORDER-E	0.9593	0.9795	**0.8024**	**0.8716**

JORDER-R-DEVEIL (Section 5.2 of [2]), JORDER-E (one version with the improvements presented in our paper), JORDER-E-DEVEIL (JORDER-E + detail preserving rain-accumulation removal) with state-of-the-art methods: image decomposition (ID) [12], CNN-based rain drop removal (CNN) [38], discriminative sparse coding (DSC) [15], layer priors (LP) [21], deep detail network (DetailNet) [1], directional global sparse model (UGSM) [63], joint convolutional analysis and synthesis sparse representation (JCAS) [64], density-aware multi-stream dense network (DID-MDN) [65], conditional generative adversarial network (ID-CGAN) [66], and a common CNN baseline for image processing – SRCNN [25]. All our methods, SRCNN, and DetailNet are trained from scratch. Other methods come from online available resources kindly provided by the authors. For evaluations on synthesized data, we train the model with the corresponding training data from scratch, without any fine-tuning.

For the experiments on synthesized data, two metrics Peak Signal-to-Noise Ratio (PSNR) [67] and Structure Similarity Index (SSIM) [68] are used as comparison criteria. We evaluate the results only in the luminance channel, which has a significant impact on the human visual system to perceive the image quality. Our results and codes are publicly available.

Implementation Details. JORDER uses 20 layers as its standard setting. The skip connections are set with an interval of 2 convolution layers. The number of channels in each convolution layer is fixed as 64. The training and testing images are cropped into small sub-images with a size of 48×48 pixels. We use flipping (up-down and left-right) for data augmentation. For each training image, three augmented images are generated. The final training set contains around 1,200,000 sub-images for rain-streak and rain-accumulation removal, respectively. Empirically, λ_1 is set as 0.01, and λ_2 is set as 0.001, receptively. The losses of rain detection and estimation are auxiliary for decreasing rain removal loss.

We train our model on Caffe [69]. Stochastic gradient descent (SGD) is used for training the model. In particular, we set the momentum as 0.9, the initial learning rate as 0.001 and change it to 0.0001 after 81 epochs, and to 0.00001 after 108 epochs. We only allow at most 135 epochs. The learning rate of the last convolution layer is set 0.01 times the global one during the whole training process. The transmission is randomly sampled from a uniform distribution with a range $[0.5, 1]$, and γ used for synthesizing low light images is randomly sampled from a uniform distribution with a range $[1.0, 1.5]$.

Quantitative Evaluation. Tables 1 and 2 show the results of different methods. As observed, our method considerably outperforms other methods in terms of both PSNR and SSIM on *Rain12*, *Rain100L*, *Rain100H* and *Rain800*. Our JODDER-R achieves considerably better results than the other methods. The PSNR of JORDER-R gains over JORDER more than 1dB on *Rain100H*. Such a large gain demonstrates that the recurrent rain detection and removal significantly boosts the performance on synthesized heavy rain images. Principally, compared with heavy rain cases, in normal rain, the rain streaks are sparser, and the rain accumulation is less dense. Thus, it is easier to remove normal rains than heavy rains. Our rain model also is capable of synthesizing normal rain images, and our training set includes those images. Hence, our method, which is designed to handle heavy rains, can also be generalized to handle normal rains, and achieves good results.

Qualitative Evaluation. Fig. 6 shows the results of real images. For fair comparisons, we use JORDER-R to process these rain images and do not handle rain accumulation on these results, to be consistent with other methods. As observed, our method significantly outperforms them and is successful in removing the majority of rain streaks. We also compare all the methods in two extreme cases: dense rain accumulation, and heavy rain as shown in Fig. 7. Our method achieves promising results in removing the majority of rain streaks, enhancing the visibility and preserving details.

Running Time. Table 3 compares the running time of several state-of-the-art methods. All baseline methods are implemented in MATLAB. Our methods are implemented on the Caffe's Matlab wrapper. CNN rain drop and some versions of our methods are implemented on GPU, while others are based on CPU. Our GPU versions is computationally efficient. The CPU version of JORDER, a lightest version of our method, takes up the shortest running time among all CPU-based approaches. In general, our methods in GPU are capable of dealing with a 500×500 rain image less than 10s.

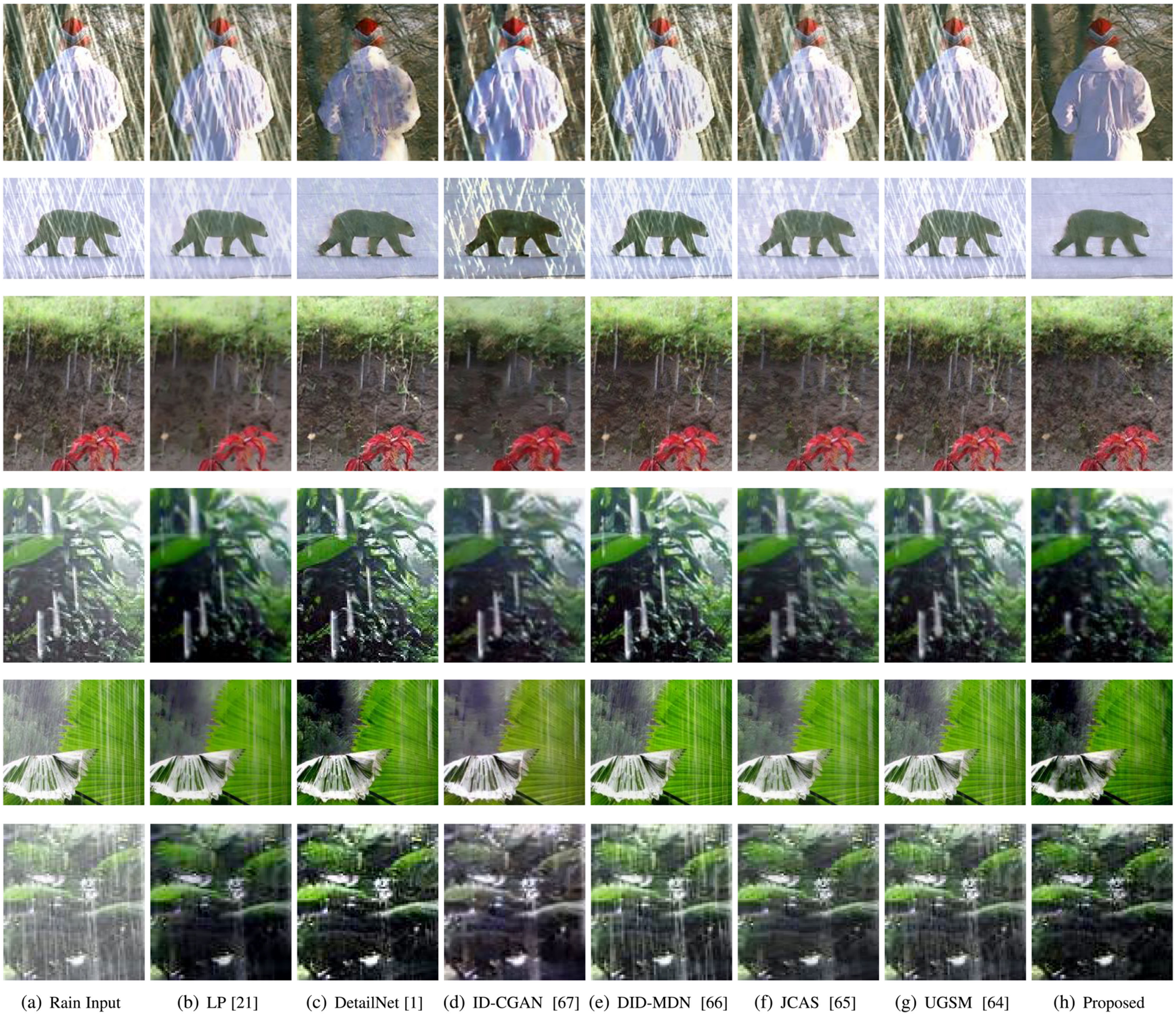


Fig. 6. Results of different methods on synthesized and real images. Zooming in the images will show that our method is superior to others. The 1st-2st panels: synthesized rain images. The 2rd-4th panels: real rain images.

Features Visualization for First and Last Layers. To have a glimpse on what happening in our network, we visualize the feature maps produced by our network. Fig. 8 shows the feature maps of the first and last convolution layers. The 24 feature maps with the highest responses, measured by variances of the feature maps, are presented. Fig. 8b clearly shows that the first convolution behaves like calculating image gradients, where interestingly the texture details that are uncorrelated to the rain streaks are preserved. From the visualization results, many grass details appear in these feature maps. That is to say, in the shallow layers, the image details including rain streaks and normal textures are both extracted from the input rain image. After the transformations by the middle layers of the network, the feature maps in the last layer are highly correlated with rain streaks and their context, as shown in Fig. 8c. For some feature maps, only rain streaks are observed, as shown in the red blocks of Fig. 8c. Although some normal textures are also included as shown in the blue blocks of Fig. 8c, these textures are consistent within a map. It demonstrates that the network plays a role of texture separation from shallow layers to deep layers.

Evaluation on Texture Preservation. We compare the texture preserving capacity among rain removal methods by computing the absolute differences between the groundtruths and the predicted clean background. Fig. 9 clearly shows that, DetailNet and JORDER are significantly better than LP [21], DID-MDN [65], ID-CGAN [66], JCAS [64], and UGSM [63]. Compared to DetailNet, JORDER generates more sparse results and only has large responses in the texture regions along the vertical directions.

Evaluations on Streak and Rain-Accumulation Removal. Regarding the order of rain-streak removal and rain-accumulation removal, we compare a few different combinations: (1) rain-streak removal alone, or derain, (2) rain-streak removal twice, derain-derain, (3) rain-streak removal followed by rain-accumulation removal, derain-deveil, (4) rain-accumulation removal followed by rain-streak removal, deveil-derain, and finally (5) our proposed order, derain-deveil-derain. Fig. 10 shows the comparison results.

Evaluation on Detail Preserving Rain-Accumulation Removal. We compare the results of different rain-accumulation methods. As shown in Fig. 11, without low light degradation in training data generation, the rain removal results

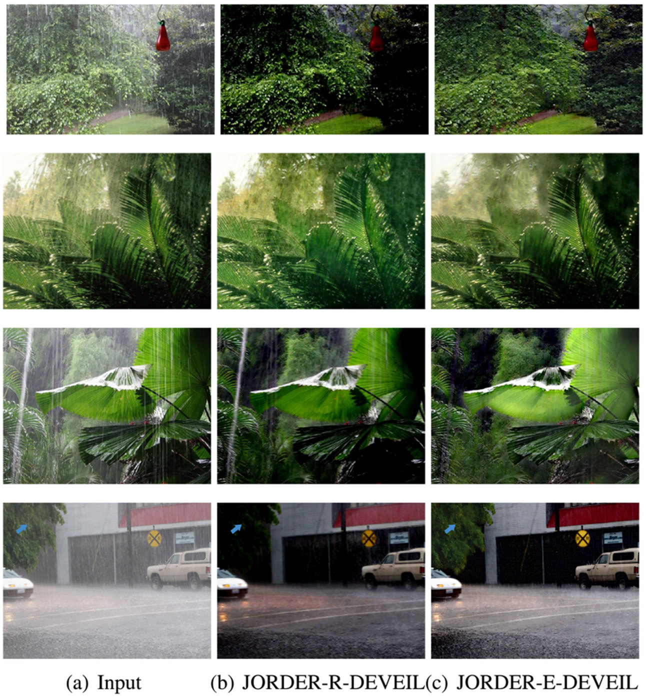


Fig. 7. Examples of our method on heavy rain and mist images.

become darker and some details turn invisible (R in Fig. 11)). Training with low light degradation enhances visibility of dark details (From R to L). The added white balance constraint makes the rain removal results look more naturally in colors (From L to L+W).

Computer Vision Applications. While our method provides visually pleasing results, it can also improve computer vision tasks. Here, we experiment with image classification with and without rain removal. We synthesize 50,000, 2,000, and 4,952 rain images with the validation set of ImageNet-1k dataset, the validation set of ADE20K Dataset [70], and the testing set of VOC 2007 [71] for object classification, semantic segmentation, and object detection. The results of object classification, semantic segmentation, and object detection with and without rain removal are shown in Table 4 and 5. The top-1, top-5, top-10 accuracies are used as metrics and VGG-19 [72] is used as the classification model. Mean IOU and accuracy are used as the metrics of semantic segmentation, and Mean AP is used as the metric of object detection. As can be observed from the results, our rain removal significantly boosts the performance on all metrics and push them to get close to their rain-free ones.

We also show two cases of applying our method as preprocessing for a commercial computer vision system, Clarifai,[2] which is an advanced image recognition system based on a deep convolutional network. The two images are shown in Fig. 12. Before rain removal, these images are inaccurately categorized as 'Rain' and 'Nature'. After rain removal by our method, they are labeled correctly as 'People'.

Comparison to State-of-the-art Dehazing Methods. To demonstrate the superiority of our rain accumulation removal method, we compare the proposed method with other state-of-the-art dehazing methods in Fig. 13. As one can observe, the result of Nonlocal has color shift. DehazeNet and GFN tend to produce darker results and retains some accumulation. Comparatively, our rain accumulation removal method successfully removes most accumulation and lights up dark details in the images.

TABLE 3
The Time Complexity (in Seconds) of JORDER Compared with State-of-the-Art Methods. JR and JRD Denote JORDER-R and JORDER-R-DEVEIL, Respectively

Baseline	CNN (G) [38]	DSC (C) [15]	LP (C) [21]
80×80	449.94	14.32	35.97
500×500	1529.85	611.91	2708.20
Baseline	DetailNet (G) [1]	UGSM (C) [64]	JCAS (C) [65]
80×80	0.02	0.09	2.59
500×500	0.58	2.3	179.56
Baseline	JORDER (C)	JORDER (G)	JR (G)
80×80	2.97	0.11	0.32
500×500	69.79	1.46	3.08
Baseline	JRD (G)	DID-MDN (G) [66]	ID-CGAN (G) [67]
80×80	0.72	0.56	0.03
500×500	7.16	2.94	0.57

(G) and (C) denote the implementation on GPU and CPU, respectively.

Case Studies for Rain Types on Rain Removal Performance. We compare four types of rains: occluding rain streaks, rain accumulation, veiling rain streaks, and sparkle rain streaks in Fig. 14. It is observed from the results that, our method successfully removes most of occluding rain streaks, rain accumulation, and veiling rain streaks. For sparkle rain streaks, ID-CGAN, DID-MDN and LP sometimes achieve superior performance to our JORDER network. The main reason is that, our training set includes small little sparkle rain streaks, and our JORDER network is not imposed with any smoothness constraint, such as total variation. In the future, we will try to incorporate local smoothness constraints, such as nonlocal regularization terms or low rank priors, into our JORDER network to make it perform better in the cases with sparkle rain streaks.

The Effect of Rain Models and Training Datasets. We compare several JORDER networks trained with different datasets: *rain100L*, *rain100H* and *rain800*. The first one is synthesized with single-layer rain streaks. The second is

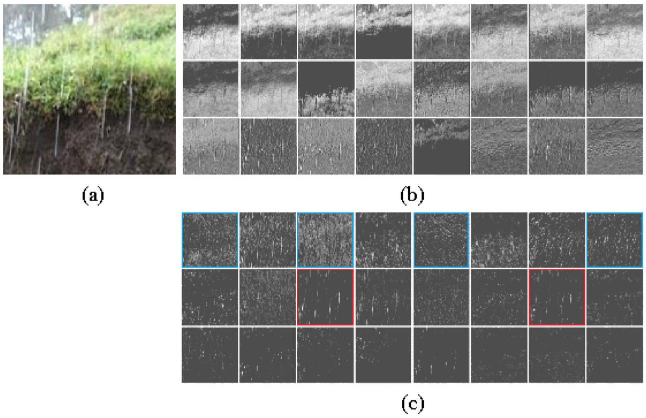


Fig. 8. Visualization of features in the first and last convolution layers for a 150×150 sub-image. (a) The input region. (b) The 24 feature maps with the highest responses in the first layer. (c) The 24 feature maps with the highest responses in the last layer.

[2] https://www.clarifai.com/.

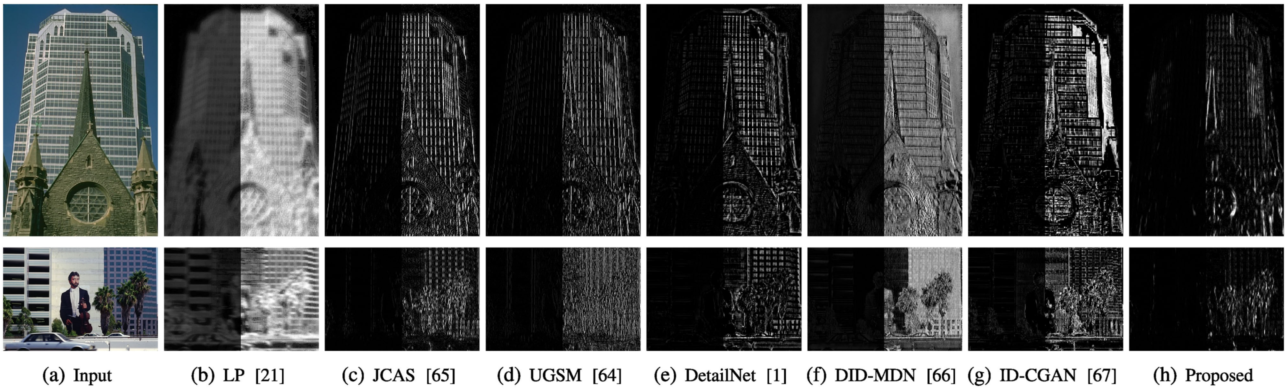


Fig. 9. Texture preservation comparison. The left half part of each image is the absolute difference between the background prediction and the ground truth image. In the right half part, the value is enlarged by a factor of 5 for better observation.

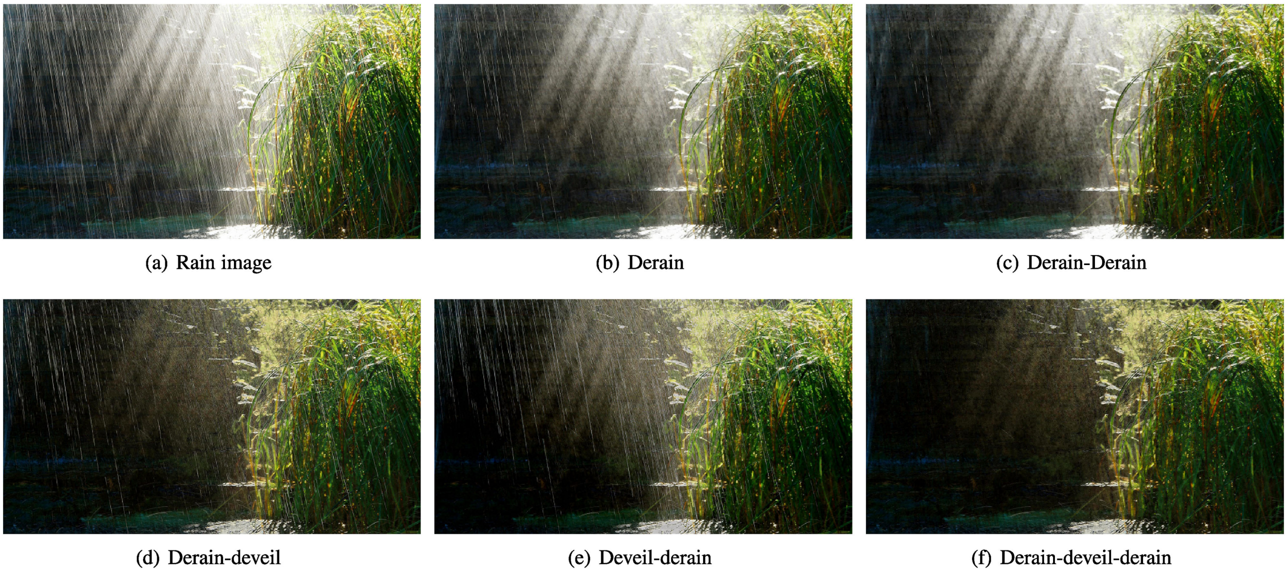


Fig. 10. The results of JORDER-R-DEVEIL in different orders.

synthesized with multi-layer rain streaks. The streaks used in these two cases are synthesized based on a raindrop oscillation model [5]. The last one is synthesized from random noises with different noise levels. As one can observe, the synthesized training data plays an important role in removing rains in real cases. The JORDER network trained on *Rain100H* achieves significantly superior rain streak removal performance than those trained on *Rain100L* and *Rain800*, which demonstrate the effectiveness of our rain synthesis model. The results generated by the model trained on *Rain100L* have many residual streaks. The results generated by the model trained on *Rain800* may lead to unexpected light changes, as shown in the last row of Fig. 15.

Evaluations of Different Network Architectures. Concerning the choice of our architecture (Fig. 2), we also compare a few different possible architectures, where all of them intend to estimate $\mathbf{B}, \mathbf{S}, \mathbf{R}$:

$$\arg\min_{\mathbf{B},\mathbf{S},\mathbf{R}} ||\mathbf{O} - \mathbf{B} - \mathbf{S}\mathbf{R}||_2^2 + P_b(\mathbf{B}) + P_s(\mathbf{S}) + P_r(\mathbf{R}). \tag{11}$$

Principally we have two choices for the network structures as shown in Fig. 16: parallel and sequential. Specifically, considering the prediction order of the variables, there are three candidates:

1) Sequential structure 1: Predicting $\mathbf{R}$ followed by $\mathbf{S}$ and then $\mathbf{B}$, which is denoted as RSB and our final proposed architecture

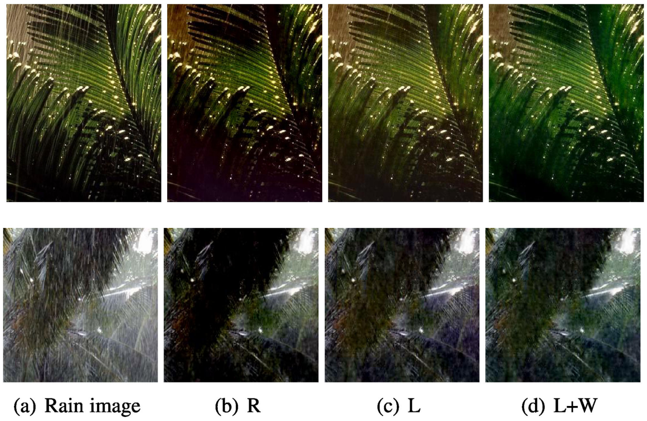


Fig. 11. The results with and without the detail preserving rain-accumulation removal. R denotes a raw model, trained without low light degradation and white balance constraint. L denotes the version trained with low light degradation. L+W denotes the version trained with low light degradation and white balance constraint.

TABLE 4
The Error Rate of VGG-19 with / without Rain Removal as a Preprocessing on ImageNet-1k Validation Dataset

Metric	top-1(%)	top-5(%)	top-10(%)
Without Streaks	66.15	86.95	91.53
Without Rain Removal	43.53	67.09	75.12
With Rain Removal	60.89	83.16	88.73

TABLE 5
The Semantic Segmentation and Object Detection Performance of Pretrained Models with / without Rain Removal as a Preprocessing on *MIT ADE20K* and *VOC 2007* Validation Dataset

Metric	Mean IOU	Accuracy (%)	Mean AP
Dataset	*MIT ADE20K*		*VOC 2007*
Without Streaks	0.4063	79.63	0.7014
Without Rain Removal	0.3746	77.61	0.6618
With Rain Removal	0.2675	67.95	0.5755

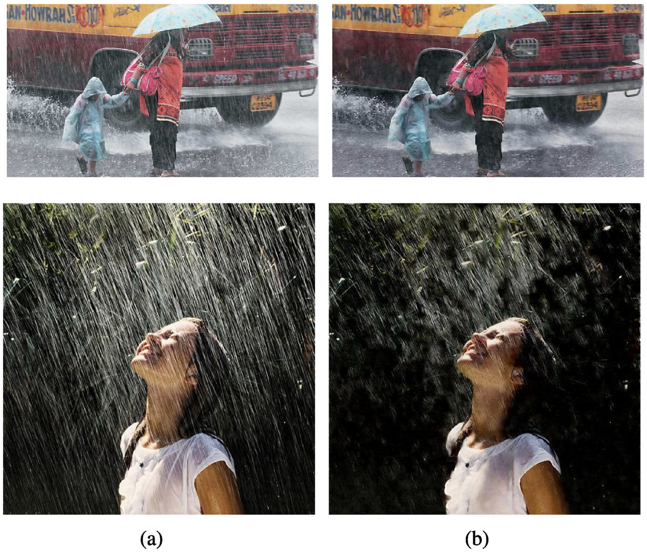


Fig. 12. Image recognition results on the images before and after rain-streak removal. Top panel: (a) Before, labeled as ‘Rain’. (b) After, labeled as ’People’. Bottom panel: (a) Before, labeled as ‘Nature’. (b) After, labeled as ‘People’.

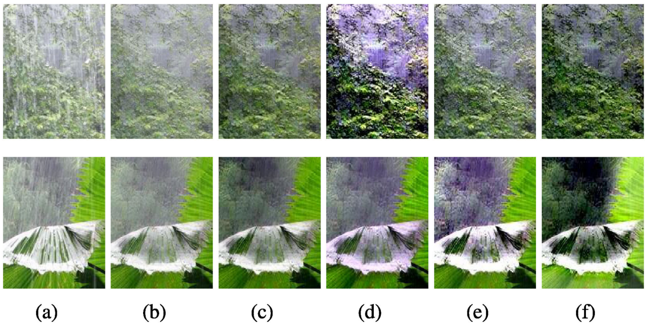
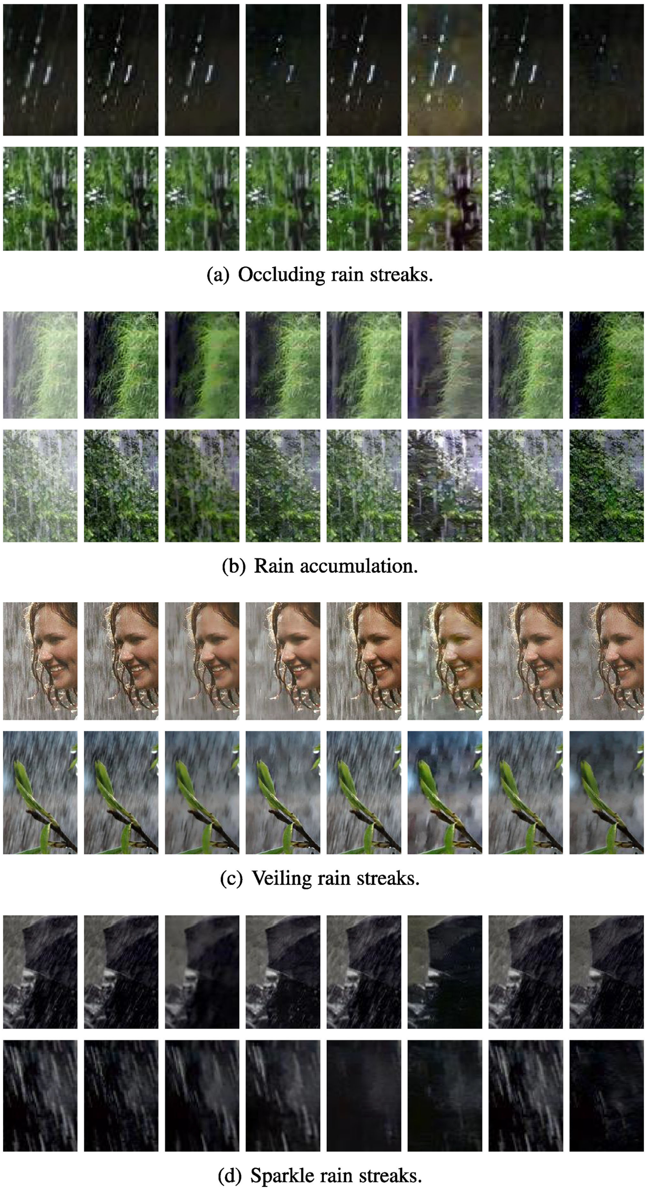


Fig. 13. Visual comparison of our rain accumulation removal with state-of-the-art dehazing algorithms on real rain images with rain streak accumulation. (a) Input. (b) JORDER-R. (c) DehazeNet [36]. (d) Nonlocal [74]. (e) GFN [75]. (f) Proposed. All methods take the rain streak removal results produced by our JORDER-R as their inputs. It is observed that, our rain accumulationd removal method is successful in removing most accumulation and lighting up details in dark regions.



(a) Occluding rain streaks.

(b) Rain accumulation.

(c) Veiling rain streaks.

(d) Sparkle rain streaks.

Fig. 14. Visual results of different methods on four types of rains. (a) Occluding rain streaks. (b) Rain accumulation. (c) Veiling rain streaks. (d) Sparkle rain streaks. From left to right: input rain image, DSC [15], LP [21], DetailNet [1], DID-MDN [66], ID-CGAN [67], UGSM [64], Proposed.

2) The parallel structure: Predicting **S** and **R** in parallel based on **F**), denoted as PAR
3) Sequential structure 2: Predicting **S** followed by **R** and then **B** in order), denoted as SRB
4) Vanilla ResNet: A four-layer ResNet with only one recurrence to directly predict the background image, denoted as RAW

Note that, all the experiments here do not include the contextualized dilated convolution.

We compare the training performance and objective quality of these possible architectures on *Rain20L* with PSNR and SSIM as the evaluation metrics as shown in Fig. 19 and in Table 6. The experimental results clearly show the superiority of PAR and RSB.

Ablation Study for Contextualized Dilated Convolution. We look further into the benefit of the contextualized dilated

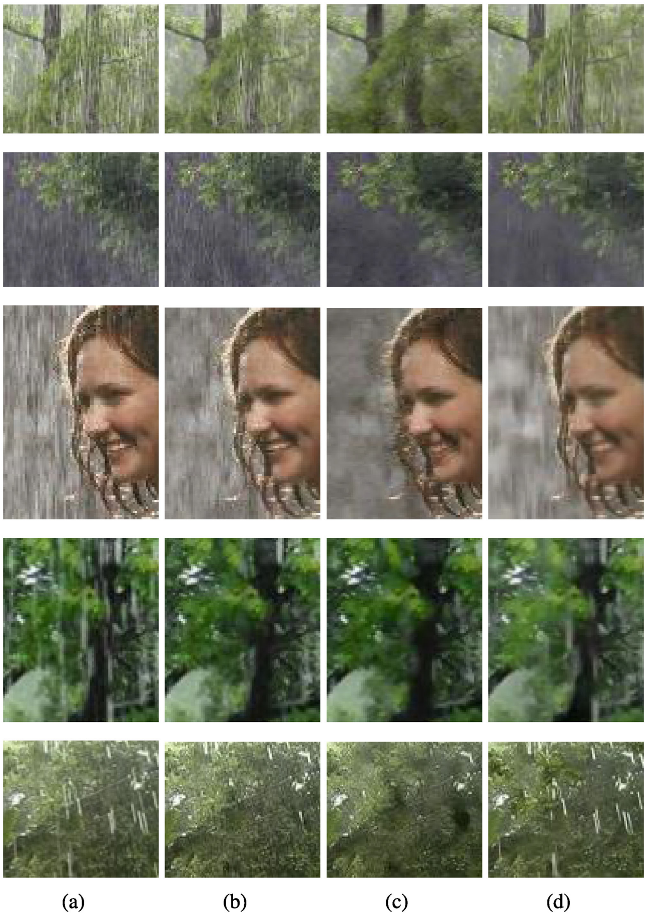


Fig. 15. Comparison of rain removal results generated by the models trained on different training sets. (a) Input images. (b) Results generated by the network trained on *Rain100L*. (c) Results generated by the network trained on *Rain100H*. (c) Results generated by the network trained on *Rain800*.

convolution to the final performance. Three coupled versions are involved in the comparisons: PARD, which is boosted PAR with the contextualized dilated convolution; RSBD, which is boosted RSB with the contextualized dilated convolution; and, JORDER- (10-layer RSB), which is JORDER (10-layer RSBD) without the contextualized dilated convolution. The training performance is shown in Figs. 17 and 18. The quantitative comparison is shown in Table 7. The experimental results clearly demonstrate the positive effect of the contextualized dilated convolution on the final objective performance. These results suggest a valuable conclusion: RSBD structure is better at acquiring contextual information than PARD. This is the reason why our final choice of architecture is RSBD (Fig. 2).

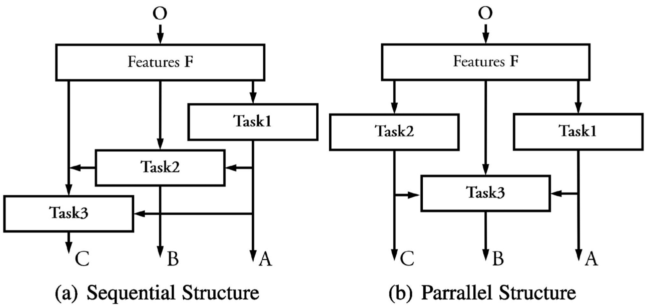


Fig. 16. Potential choices for network architectures.

TABLE 6
PSNR and SSIM Results of the Four Versions

Baseline	RSB	PAR	SRB	RAW
PSNR	34.66	34.63	34.28	33.94
SSIM	0.9622	0.9624	0.9593	0.9576

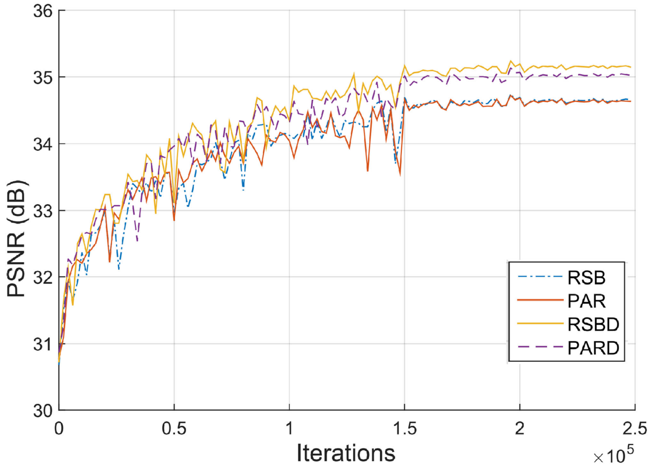


Fig. 17. The PSNR result with and without contextualized convolutions in the training process. We drop the learning rate from 0.001 to 0.0001 when reaching 1.5×10^5 iterations and from 0.0001 to 0.00001 when reaching 2×10^5 iterations.

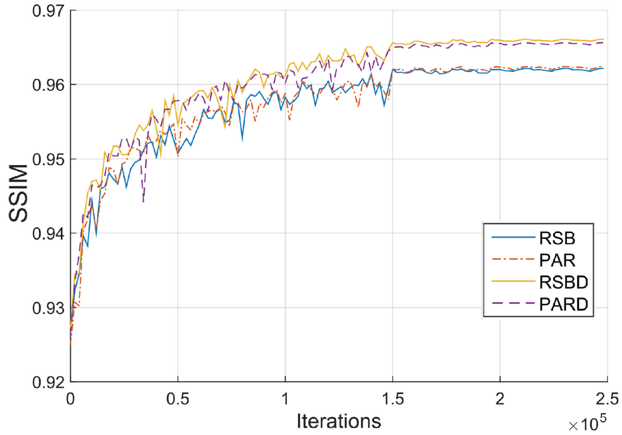


Fig. 18. The SSIM result with and without contextualized convolutions in the training process. We drop the learning rate from 0.001 to 0.0001 when reaching 1.5×10^5 iterations and from 0.0001 to 0.00001 when reaching 2×10^5 iterations.

TABLE 7
Objective Evaluation for the Effect of Contextualized Dilated Convolution

Baseline	PAR	PARD	RSB	RSBD
Dataset	*Rain20L*		*Rain20L*	
PSNR	34.63	35.06	34.66	35.16
SSIM	0.9624	0.9655	0.9622	0.966
Baseline	JORDER-	JORDER	JORDER-	JORDER
Dataset	*Rain100L*		*Rain100H*	
PSNR	35.41	36.11	20.79	22.15
SSIM	0.9632	0.9711	0.5978	0.6736

Evaluations in Potential Architectures of Contextualized Dilated Convolutions. The performance of JORDER network with and without contextualized dilated convolutions (CDC)

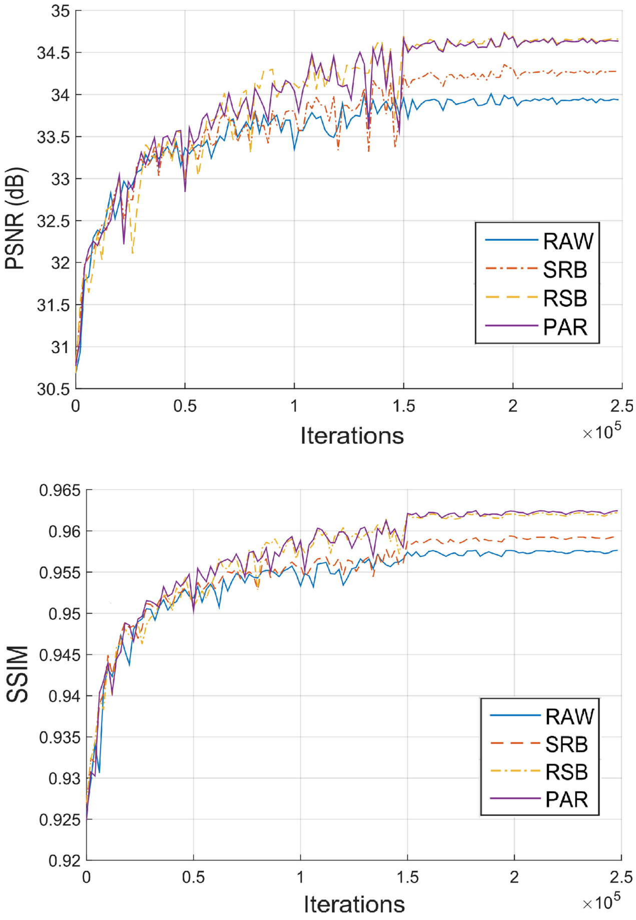


Fig. 19. The training performance of the four networks. We drop the learning rate from 0.001 to 0.0001 when reaching 1.5$\times10^5$ iterations and from 0.0001 to 0.00001 when reaching 2$\times10^5$ iterations.

is provided in Table 8. The parallel network signifies the version illustrated in Fig. 3, and the sequential network signifies that the convolution paths with different dilated factors are chained together. It is observed that, adding contextualized dilated convolutions significantly improves the rain removal performance. Compared with the parallel architecture, the sequential architecture further provides a performance gain.

TABLE 8
The Performance of JORDER Network with and w/o Contextualized Dilated Convolutions (CDC)

Baseline	Metric	*Rain100L*	*Rain100H*	*Rain800*
w/o CDC	PSNR	35.41	20.79	25.61
CDC (parallel)		36.11	22.15	26.03
CDC (sequential)		36.37	22.45	26.23
w/o CDC	SSIM	0.9632	0.5978	0.8378
CDC (parallel)		0.9741	0.6736	0.8501
CDC (sequential)		0.9767	0.6972	0.8575

The parallel one is illustrated in Fig. 3, and the sequential one signifies that the convolution paths with different dilated factors are chained together.

We also evaluate the performance of JORDER networks in Table 9 where dilated convolutions are replaced by pooling and up-sampling layers (JPS), and stride convolution and transposed convolution layers (JST), respectively. It is observed that, the multi-path pooling and up-sampling layers (JPS) lead to a performance drop. The stride convolution and transposed convolution layers (JST) can also improve the objective metrics, and only lead to an inferior performance to JORDER, which demonstrates the superior capacity of our JORDER to obtain context while keeping rich local details.

Visualization of Side Features W/ and W/O Rain Detection. We visualize the side features with and without a rain detection loss in Fig. 20. It is observed that, the information captured by automatic learned side features is mostly correlated to the rain-free context, which demonstrates the necessity of an imposed rain detection loss in our JORDER network.

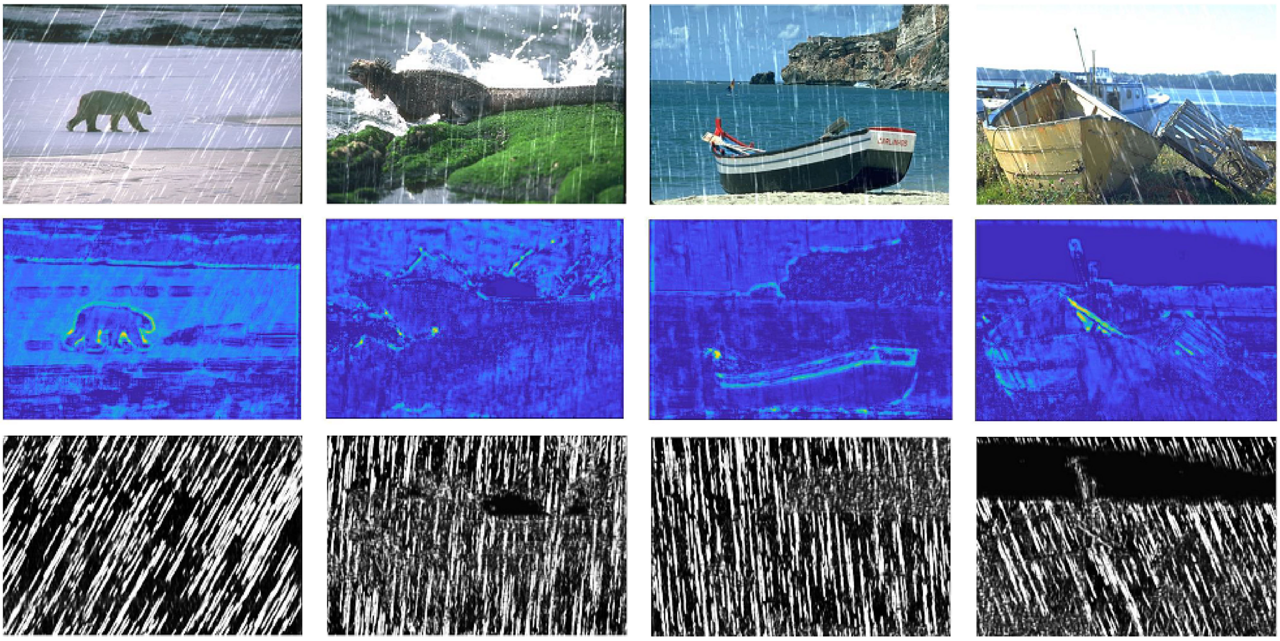


Fig. 20. Comparison of the learned features with and without a rain detection loss. (a) Input rain images. (b) Rain features without a rain detection loss. (c) Detected rain masks in JORDER.

TABLE 9
The Performance of JORDER Networks Where Dilated Convolutions Are Replaced by Pooling and Up-Sampling Layers (JPS), and Stride Convolution and Transposed Convolution Layers (JST)

Baseline	*Rain12*	*Rain100L*	*Rain100H*	*Rain800*
Metric		PSNR		
JORDER-	35.86	35.41	20.79	25.61
JPS	35.62	35.11	20.53	25.42
JST	35.93	35.82	21.67	25.77
JORDER	36.02	36.11	22.15	26.03
Metric		SSIM		
JORDER-	0.9534	0.9632	0.5978	0.8378
JPS	0.9511	0.9584	0.5682	0.8212
JST	0.9589	0.9764	0.6713	0.8519
JORDER	0.9644	0.9820	0.7490	0.8683

TABLE 10
The Objective Evaluation When the Detected Rain Masks Are Inaccurate

Error Rate	0%		25%		50%	
Metric	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Rain100L	36.11	0.9741	35.88	0.9671	35.68	0.9610
Rain100H	22.15	0.6736	21.78	0.6378	21.48	0.6023
Rain800	26.03	0.8501	25.81	0.8457	25.63	0.8420

TABLE 11
The PSNR Results in the Case of Over-Detection and Under-Detection

Baseline	Original	Exp-10%	Exp-20%	Exp-30%
Rain100L	36.11	36.01	35.90	35.85
Rain100H	22.15	21.99	21.82	21.71
Rain800	26.03	25.87	25.78	25.76
Baseline	/	Shrink-10%	Shrink-20%	Shrink-30%
Rain100L	/	36.00	35.91	35.82
Rain100H	/	22.02	21.78	21.66
Rain800	/	25.86	25.79	25.77

Robustness Analysis of Inaccurate Rain Detection. We evaluate the effect of inaccurate rain detection on the final performance. We randomly invert the detected rain mask and calculate their PSNR and SSIM results in Table 10. As one can observe that, inverting the detected rain mask leads to a performance drop. When 50 percent rain mask pixels are inverted, the PSNR and SSIM of our JORDER drop by a margin of 0.57 dB, 0.67 dB, 0.40 dB and 0.0131, 0.0713, 0.0081 on *Rain100L*, *Rain100H*, *Rain800*, respectively, which indicates that the detected rain mask indeed plays a guidance role in rain streak removal of JORDER network.

Analysis of Over-Detection and Under-Detection of Rains. We evaluate the effect of over-detection and under-detection of rains on the final performance. We use dilation and erosion operations to dilate and erode the detected rain mask. The evaluation results are presented in Tables 11 and 12. The numbers after Exp and Shrink denote the changed percentage of areas after dilation and erosion operations, respectively. As one can observe that, over-detection and under-detection lead to a performance drop, which indicates that the detected rain mask indeed plays a guidance role in rain streak removal of JORNER network.

TABLE 12
The SSIM Results in the Case of Over-Detection and Under-Detection

Baseline	Original	Exp-10%	Exp-20%	Exp-30%
Rain100L	0.9741	0.9712	0.9683	0.9651
Rain100H	0.6736	0.6501	0.6412	0.6254
Rain800	0.8501	0.8470	0.8463	0.8448
Baseline	/	Shrink-10%	Shrink-20%	Shrink-30%
Rain100L	/	0.9716	0.9686	0.9646
Rain100H	/	0.6500	0.6410	0.6257
Rain800	/	0.8472	0.8464	0.8449

TABLE 13
Ablation Analysis for the Technical Improvements Compared to JORDER-R [2]

Baseline	JORDER-R	RTL	CF-MSL	L_1
PSNR	23.45	24.39	24.43	24.54
SSIM	0.7490	0.7992	0.7987	0.8024

TABLE 14
Comparison of the Rank Product of Different Methods

Method	LP	DetailNet	ID-CGAN	DID-MDN
Rank	5.25	3.54	4.46	4.41
Method	DSC	JCAS	UGSM	Proposed
Rank	7.07	2.92	5.35	1.00

The smaller, the better.

Ablation Analysis for Our Improvements. We perform an ablation analysis for the components added in this paper on *Rain100H* compared with our previous work [2] in Table 13. RTL denotes the residual task learning (Section 5.1). CF-MSL denotes the coarse-to-fine multi-scale loss (Section 4.3). L_1 denotes to use L_1 norm of the difference instead of MSE as the training loss (Section 4.3). Every component is added into the network from left to right. The results demonstrate the effectives of each component in our paper. Residual task learning boosts the performance a lot. Coarse-to-fine multi-scale loss and L_1 further improve our performance compared to pure L_2 loss.

User Study in Subjective Evaluation. We employ paired comparison approach, where the participants are shown two results at a time and are asked to simply choose the preferred one by rain removal quality. The rank product [75] results are presented in Table 14. We have a total of 25 participants, including both domain experts and generally knowledgeable individuals, each given 252 pairwise comparisons over a set of eight testing images with eight different rain removal methods. Compared with others, the proposed method achieves the best visual quality.

Analysis for Recurrence Number of Contextualized Dilated Convolutions. We analyze the effect of the recurrence number of contextualized dilated convolutions on the deraining performance. The results are presented in Figs. 21 and 22. It can observed that, higher PSNR and SSIM results are obtained when more recurrences are used. The marginal PSNR and SSIM gains converge when the recurrence number reaches 9.

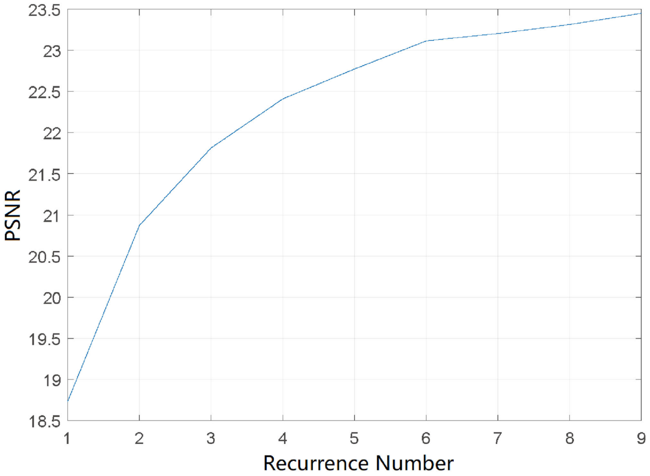


Fig. 21. PSNR performance curve for networks with different recurrences.

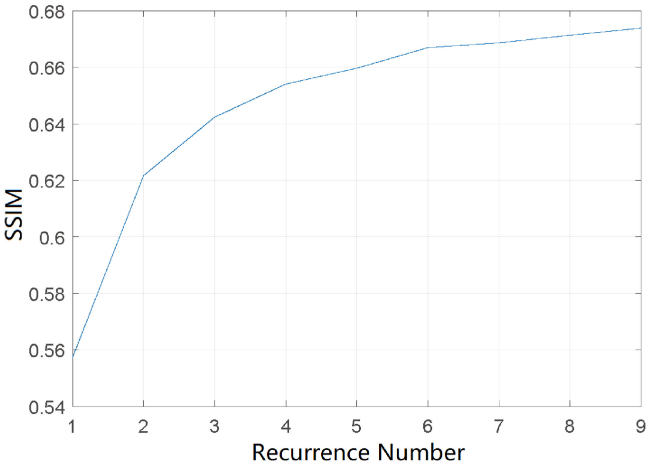


Fig. 22. SSIM performance curve for networks with different recurrences.

7 Conclusion

In this paper, we have introduced a new deep learning based method to remove rain from a single image, even in the presence of rain streak accumulation and heavy rain. A new region-dependent rain image model is proposed for additional rain detection and is further extended to simulate rain accumulation and heavy rains. Based on this model, we developed a fully convolutional network that jointly detect and remove rain. Rain regions are first detected by the network which naturally provides additional information for rain removal. To restore images captured in the environment with both rain accumulation and heavy rain, we introduced an recurrent rain detection and removal network that progressively removes rain streaks, embedded with the rain-accumulation removal network. Evaluations on real images demonstrated that our method outperforms state-of-the-art methods.

Acknowledgments

This work was supported by National Natural Science Foundation of China under contract No. 61772043 and Beijing Natural Science Foundation under contract No. L182002 and No. 4192025. We also gratefully acknowledge the support of NVIDIA Corporation with the GPU for this research. R.T. Tan's research is supported in part by Yale-NUS College Start-Up Grant.

References

[1] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley, "Removing rain from single images via a deep detail network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1715–1723.

[2] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1685–1694.

[3] K. Garg and S. K. Nayar, "When does a camera see rain?" in *Proc. IEEE Int. Conf. Comput. Vis.*, 2005, vol. 2, pp. 1067–1074.

[4] K. Garg and S. K. Nayar, "Detection and removal of rain from videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2004, vol. 1, pp. I–528.

[5] K. Garg and S. K. Nayar, "Photorealistic rendering of rain streaks," *ACM Trans. Graph.*, vol. 25, no. 3, 2006, pp. 996–1002.

[6] K. Garg and S. K. Nayar, "Vision and rain," *Int. J. Comput. Vis.*, vol. 75, no. 1, pp. 3–27, 2007.

[7] X. Zhang, H. Li, Y. Qi, W. K. Leow, and T. K. Ng, "Rain removal in video by combining temporal and chromatic properties," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2006, pp. 461–464.

[8] N. Brewer and N. Liu, "Using the shape characteristics of rain to identify and remove rain from video," in *Proc. Joint IAPR Int. Workshops SPR SSPR*, 2008, pp. 451–458.

[9] P. C. Barnum, S. Narasimhan, and T. Kanade, "Analysis of rain and snow in frequency space," *Int. J. Comput. Vis.*, vol. 86, no. 2–3, pp. 256–274, 2010.

[10] J. Bossu, N. Hautière, and J.-P. Tarel, "Rain or snow detection in image sequences through use of a histogram of orientation of streaks," *Int. J. Comput. Vis.*, vol. 93, no. 3, pp. 348–367, 2011.

[11] Y.-L. Chen and C.-T. Hsu, "A generalized low-rank appearance model for spatio-temporally correlated rain streaks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 1968–1975.

[12] L. W. Kang, C. W. Lin, and Y. H. Fu, "Automatic single-image-based rain streaks removal via image decomposition," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1742–1755, Apr. 2012.

[13] D.-A. Huang, L.-W. Kang, Y.-C. F. Wang, and C.-W. Lin, "Self-learning based image decomposition with applications to single image denoising," *IEEE Trans. Multimedia*, vol. 16, no. 1, pp. 83–93, Jan. 2014.

[14] S.-H. Sun, S.-P. Fan, and Y.-C. F. Wang, "Exploiting image structural similarity for single image rain removal," in *Proc. IEEE Int. Conf. Image Process.*, 2014, pp. 4482–4486.

[15] Y. Luo, Y. Xu, and H. Ji, "Removing rain from a single image via discriminative sparse coding," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 3397–3405.

[16] L. Shen, Z. Yue, Q. Chen, F. Feng, and J. Ma, "Deep joint rain and haze removal from single images," ArXiv e-prints: 1801.06769, Jan. 2018.

[17] J.-H. Kim, C. Lee, J.-Y. Sim, and C.-S. Kim, "Single-image deraining using an adaptive nonlocal means filter," in *IEEE Trans. Image Process.*, 2013, pp. 914–917.

[18] D.-A. Huang, L.-W. Kang, M.-C. Yang, C.-W. Lin, and Y.-C. F. Wang, "Context-aware single image rain removal," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2012, pp. 164–169.

[19] V. Santhaseelan and V. K. Asari, "Utilizing local phase information to remove rain from video," *Int. J. Comput. Vis.*, vol. 112, no. 1, pp. 71–89, Mar. 2015. [Online]. Available: https://doi.org/10.1007/s11263-014-0759-8

[20] J. H. Kim, C. Lee, J. Y. Sim, and C. S. Kim, "Single-image deraining using an adaptive nonlocal means filter," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2013, pp. 914–917.

[21] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown, "Rain streak removal using layer priors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2736–2744.

[22] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley, "Clearing the skies: A deep network architecture for single-image rain removal," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2944–2956, Jun. 2017.

[23] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.

[24] S. Iizuka, E. Simo-Serra, and H. Ishikawa, “Globally and locally consistent image completion,” *ACM Trans. Graph.*, vol. 36, no. 4, 2017, Art. no. 107.

[25] C. Dong, C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 295–307, Feb. 2016.

[26] Y. Hu, J. Liu, W. Yang, S. Deng, L. Zhang, and Z. Guo, “Real-time deep image super-resolution via global context aggregation and local queue jumping,” in *Proc. IEEE Visual Commun. Image Process.*, Dec. 2017, pp. 1–4.

[27] W. Yang, S. Xia, J. Liu, and Z. Guo, “Reference guided deep super-resolution via manifold localized external compensation,” *IEEE Trans. Circuits Syst. Video Technol.*, pp. 1–1, 2018.

[28] S. Xia, W. Yang, J. Liu, and Z. Guo, “Dual recovery network with online compensation for image super-resolution,” in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2018, pp. 1–5.

[29] W. Yang, J. Feng, G. Xie, J. Liu, Z. Guo, and S. Yan, “Video super-jresolution based on spatial-temporal recurrent residual networks,” *Comput. Vis. Image Understanding*, vol. 168, pp. 79–92, 2018.

[30] W. Yang, S. Deng, Y. Hu, J. Xing, and J. Liu, “Real-time deep video SpaTial resolution UpConversion SysTem (STRUCT++ demo),” in *Proc. 25th ACM Int. Conf. Multimedia*, 2017, 1255–1256.

[31] C. J. Schuler, M. Hirsch, S. Harmeling, and B. Schölkopf, “Learning to deblur,” in *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 7, pp. 1439–1451, July 2016.

[32] L. Xu, J. S. Ren, C. Liu, and J. Jia, “Deep convolutional neural network for image deconvolution,” in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2014, pp. 1790–1798.

[33] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image Style Transfer Using Convolutional Neural Networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, 2016, pp. 2414–2423.

[34] C. Dong, Y. Deng, C. Change Loy, and X. Tang, “Compression artifacts reduction by a deep convolutional network,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 576–584.

[35] X. Zhang, W. Yang, Y. Hu, and J. Liu, “DMCNN: Dual-domain multi-scale convolutional neural network for compression artifacts removal,” in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2018, pp. 390–394.

[36] B. Cai, X. Xu, K. Jia, C. Qing, and D. Tao, “Dehazenet: An end-to-end system for single image haze removal,” *IEEE Trans. Image Process.*, vol. 25, no. 11, pp. 5187–5198, Nov. 2016.

[37] W. Ren, S. Liu, H. Zhang, J. Pan, X. Cao, and M.-H. Yang, “Single image dehazing via multi-scale convolutional neural networks,” in *Proc. IEEE Eur. Conf. Comput. Vis.*, 2016, pp. 154–169.

[38] D. Eigen, D. Krishnan, and R. Fergus, “Restoring an image taken through a window covered with dirt or rain,” in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 633–640.

[39] R. Qian, R. T. Tan, W. Yang, J. Su, and J. Liu, “Attentive generative adversarial network for raindrop removal from a single image,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2482–2491.

[40] K. G. Lore, A. Akintayo, and S. Sarkar, “LLNet: A deep autoencoder approach to natural low-light image enhancement,” *Pattern Recognit*, vol. 61, pp. 650-662, 2017.

[41] C. Wei, W. Wang, W. Yang, and J. Liu, “Deep retinex decomposition for low-light enhancement,” in *Proc. British Mach. Vis. Conf.*, Mar. 2018, pp. 127–136.

[42] J. Liu, W. Yang, S. Yang, and Z. Guo, “Erase or fill? deep joint recurrent rain removal and reconstruction in videos,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3233–3242.

[43] J. Liu, W. Yang, S. Yang, and Z. Guo, “D3R-Net: Dynamic routing residue recurrent network for video rain removal,” *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 699–712, Feb. 2019.

[44] X. Zhang, W. Lin, S. Wang, J. Liu, S. Ma, and W. Gao, “Fine-grained quality assessment for compressed images,” *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1163–1175, Mar. 2019.

[45] Y. Fang, C. Zhang, W. Yang, J. Liu, and Z. Guo, “Blind visual quality assessment for image super-resolution by convolutional neural network,” *Multimedia Tools Appl.*, vol. 77, pp. 29 829–29 846, Nov. 2018.

[46] F. Jiang, W. Tao, S. Liu, J. Ren, X. Guo, and D. Zhao, “An end-to-end compression framework based on convolutional neural networks,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 10, pp. 3007–3018, Oct. 2018.

[47] S. Xia, W. Yang, Y. Hu, S. Ma, and J. Liu, “A group variational transformation neural network for fractional interpolation of video coding,” in *Proc. Data Compression Conf.*, Mar. 2018, pp. 127–136.

[48] J. Liu, S. Xia, W. Yang, M. Li, and D. Liu, “One-for-all: Grouped variation network-based fractional interpolation in video coding,” *IEEE Trans. Image Process.*, vol. 28, no. 5, pp. 2140–2151, May 2019.

[49] Y. Hu, W. Yang, S. Xia, W. Cheng, and J. Liu, “Enhanced intra prediction with recurrent neural network in video coding,” in *Proc. Data Compression Conf.*, Mar. 2018, pp. 413–413.

[50] Y. Hu, W. Yang, S. Xia, and J. Liu, “Optimized recurrent network for intra prediction in video coding,” in *Proc. IEEE Visual Commun. Image Process.*, Mar. 2018, pp. 413–413.

[51] X. Lu, Z. Lin, X. Shen, R. Mech, and J. Z. Wang, “Deep multi-patch aggregation network for image style, aesthetics, and quality estimation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 990–998.

[52] S. G. Narasimhan and S. K. Nayar, “Vision and the atmosphere,” *Int. J. Comput. Vis.*, vol. 48, no. 3, pp. 233–254, 2002.

[53] Y. Tian and S. G. Narasimhan, “Seeing through water: Image restoration using model-based tracking,” in *Proc. IEEE Int. Conf. Comput. Vis.*, Sep. 2009, pp. 2303–2310.

[54] C. Ancuti, C. O. Ancuti, T. Haber, and P. Bekaert, “Enhancing underwater images and videos by fusion,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 81–88.

[55] W. Yang, J. Feng, J. Yang, F. Zhao, J. Liu, Z. Guo, and S. Yan, “Deep edge guided recurrent residual learning for image super-resolution,” *IEEE Trans. Image Process.*, vol. 26, no. 12, pp. 5895–5907, Dec. 2017.

[56] F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions,” in *Proc. Int. Conf. Learn. Representation*, 2016.

[57] A. Gijsenij, T. Gevers, and J. van de Weijer, “Computational color constancy: Survey and experiments,” *IEEE Trans. Image Process.*, vol. 20, no. 9, pp. 2475–2489, Sep. 2011.

[58] B. V. Funt and G. D. Finlayson, “Color constant color indexing,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 5, pp. 522–529, May 1995.

[59] T. Gevers and A. W. Smeulders, “Color-based object recognition,” *Pattern Recognit.*, vol. 32, no. 3, pp. 453–464, 1999.

[60] G. D. Finlayson, S. D. Hordley, and P. Morovic, “Colour constancy using the chromagenic constraint,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, vol. 1, pp. 1079–1086.

[61] A. Saxena, M. Sun, and A. Y. Ng, “Make3D: Learning 3D scene structure from a single still image,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 5, pp. 824–840, May 2009. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2008.132

[62] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in *Proc. IEEE Int. Conf. Comput. Vis.*, Jul. 2001, vol. 2, pp. 416–423.

[63] L.-J. Deng, T.-Z. Huang, X.-L. Zhao, and T.-X. Jiang, “A directional global sparse model for single image rain removal,” *Appl. Math. Modelling*, vol. 59, pp. 662–679, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0307904X18301069

[64] S. Gu, D. Meng, W. Zuo, and L. Zhang, “Joint convolutional analysis and synthesis sparse representation for single image layer separation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 1717–1725.

[65] H. Zhang and V. M. Patel, “Density-aware single image deraining using a multi-stream dense network,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, 2018, pp. 695-704.

[66] H. Zhang, V. Sindagi, and V. M. Patel, “Image de-raining using a conditional generative adversarial network,” ArXiv e-prints, 1701.05957, Jan. 2017.

[67] Q. Huynh-Thu and M. Ghanbari, “Scope of validity of PSNR in image/video quality assessment,” *Electron. Lett.*, vol. 44, no. 13, pp. 800–801, 2008.

[68] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[69] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” *ACM Trans. Multimedia*, 2014, pp. 675–678. [Online]. Available: http://doi.acm.org/10.1145/2647868.2654889

[70] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, “Scene parsing through ADE20K dataset,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5122–5130.

[71] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, Jan. 2015.

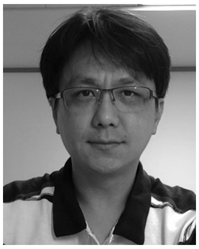
[72] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015.

[73] D. Berman, T. Treibitz, and S. Avidan, "Non-local image dehazing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogniti.*, Jun. 2016, pp. 1674–1682.

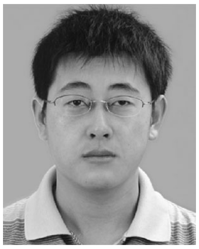
[74] W. Ren, L. Ma, J. Zhang, J. Pan, X. Cao, W. Liu, and M.-H. Yang, "Gated fusion network for single image dehazing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3253–3261.

[75] M. Rubinstein, D. Gutierrez, O. Sorkine, and A. Shamir, "A comparative study of image retargeting," *ACM Trans. Graph.*, vol. 29, pp. 160:1–160:102010.

Wenhan Yang (17'S-18'M) received the BS degree and PhD degree (Hons.) in computer science from Peking University, Beijing, China, in 2012 and 2018, respectively. He was a visiting scholar with the National University of Singapore, from 2015 to 2016. His current research interests include deep-learning based image processing, bad weather restoration, related applications and theories. He is a member of the IEEE.



Robby T. Tan received the PhD degree in computer science from the University of Tokyo. He is an associate professor at both Yale-NUS College and ECE (Electrical and Computing Engineering), National University of Singapore. Previously, he was an assistant professor at Utrecht University. His research interests include computer vision and deep learning. He is a member of the IEEE.



Jiashi Feng received the PhD degree from the National University of Singapore, in 2014. He is currently an assistant professor with the Department of Electrical and Computer Engineering, National University of Singapore. Before joining NUS as a faculty, he was a postdoc research follow at UC Berkeley. His research areas include computer vision and machine learning. In particular, he is interested in object recognition, detection, segmentation, robust learning and deep learning. He is a member of the IEEE.

Zongming Guo (M'09) received the BS degree in mathematics, and the MS and PhD degrees in computer science from Peking University, Beijing, China, in 1987, 1990, and 1994, respectively. He is currently a professor with the Institute of Computer Science and Technology, Peking University. His current research interests include video coding, processing, and communication. He is the executive member of the China-Society of Motion Picture and Television Engineers. He was a recipient of the First Prize of the State Administration of Radio Film and Television Award in 2004, the First Prize of the Ministry of Education Science and Technology Progress Award in 2006, the Second Prize of the National Science and Technology Award in 2007, the Wang Xuan News Technology Award and the Chia Tai Teaching Award in 2008, the Government Allowance granted by the State Council in 2009, and the Distinguished Doctoral Dissertation Advisor Award of Peking University in 2012 and 2013. He is a member of the IEEE.

Shuicheng Yan is currently a State Specially Recruited Expert under the Thousand Talent Program" of China and vice president of Qihoo 360 Technology Co. Ltd. and head of 360 AI Institute, responsible for leading development of four AI Engines of the company: Vision Engine, Mobility Engine, Conversation Engine and Decision Engine, which enable several business lines of the company such as smart devices, content products, commercialization and finance. He is also IEEE fellow, IAPR fellow and ACM Distinguished Scientist. His research areas include computer vision, machine learning and multimedia analysis. Till now, he has authored/co-authored about 600 high quality technical papers, with Google Scholar citation more than 40,000 times and H-index 86 (the highest among all CS researchers in Singapore). He is ISI Highly-cited researcher of 2014, 2015, 2016 and 2018. Notably, his work Network in Network (1x1 convolution) has been a standard component of almost all deep learning models in computer vision in the past few years, with great influence in both academic and industrial communities. His team was the winner of final completions of Pascal VOC 2012 and ImageNet 2017. In total, this excellent team received winner or honourable-mention prizes for more than 10 times in 8 years over these two core competitions in the computer vision field. Also they received more than 10 best paper or best student paper prizes and especially, a grand slam in ACM MM, the top conference in multimedia, including Best Paper Award, Best Student Paper Award and Best Demo Award.

Jiaying Liu (S'08-M'10-SM'17) received the BE degree in computer science from Northwestern Polytechnic University, Xi'an, China, in 2005, and the PhD degree with the best graduate honor in computer science from Peking University, Beijing, China, in 2010. She is currently an associate professor with the Institute of Computer Science and Technology, Peking University. She has authored more than 100 technical articles in refereed journals and proceedings, and holds 28 granted patents. Her current research interests include image/video processing, compression, and computer vision. She was a visiting scholar with the University of Southern California, Los Angeles, from 2007 to 2008. She was a visiting researcher at Microsoft Research Asia (MSRA) in 2015 supported by the Star Track for Young Faculties. She has also served as TC member in IEEE CAS-MSA/EOT and APSIPA IVM, and APSIPA distinguished lecturer in 2016-2017. She is a CCF/IEEE senior member.