Robust Representation Learning with Feedback for Single Image Deraining

Chenghao Chen[1] and Hao Li[*1,2]

1. Department of Automation, Shanghai Jiao Tong University (SJTU), Shanghai, 200240, China.

2. École d'Ingénieurs SJTU-ParisTech (SPEIT), Shanghai, 200240, China.

{1049164518, haoli}@sjtu.edu.cn

Abstract

A deraining network can be interpreted as a conditional generator that aims at removing rain streaks from image. Most existing image deraining methods ignore model errors caused by uncertainty that reduces embedding quality. Unlike existing image deraining methods that embed low-quality features into the model directly, we replace low-quality features by latent high-quality features. The spirit of closed-loop feedback in the automatic control field is borrowed to obtain latent high-quality features. A new method for error detection and feature compensation is proposed to address model errors. Extensive experiments on benchmark datasets as well as specific real datasets demonstrate that the proposed method outperforms recent state-of-the-art methods. Code is available at:
https://github.com/LI-Hao-SJTU/DerainRLNet

1. Introduction

Outdoor vision systems are used widely such as on intelligent vehicles and for surveillance. They sometimes suffer from rain pollution, which is undesirable in practice. To handle this problem, study on image deraining has appeared, which aims at removing rain streaks from image.

Some methods exploit specific *a priori* knowledge to clean images. For example, [8] introduces the dark channel; [7] maximizes the contrast among different target image regions. However, sparse rain streaks cannot be well removed by these methods. Some physical properties based deraining methods aim at separating the rain layer from the background layer via discriminative sparse coding [1, 20, 32], dictionary learning [14], and Gaussian mixture models [18]. However, specific *a priori* knowledge based methods are susceptible to complex, diverse, and changeable scenarios.

In recent years, deep learning based image deraining methods are rising. A deraining network can be interpreted as a conditional generator, and high-quality output images
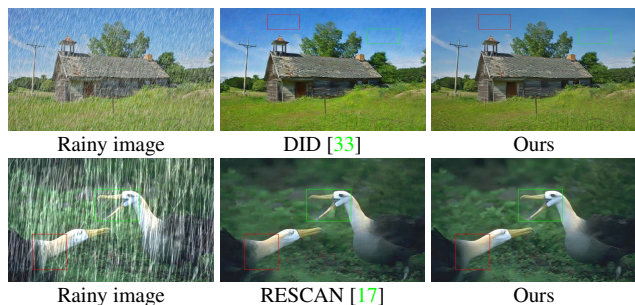
Figure 1. Image deraining. DID [33] tends to under-derain while RESCAN [17] tends to remove details of clean image.

can be generated if conditional embedding features can characterize target image contents [24]. Most deep learning based deraining methods focus on designing novel network structures and guided features, such as residual based DDN [6], density based DID [33], recurrent structure based RESCAN [17], which can be regarded as strategies for embedding enhancement. Furthermore, [24] embeds mixed feature layers into the model and decode it into a clean image. Methods that do not consider uncertainty-induced model errors tend to loss details and incur halo artefacts in the generated image (see Fig. 1).

To handle uncertainty-caused model errors, we propose a robust representation learning with feedback for image deraining. Given an image with rain streaks, the objective is to output the residual map. Then the clean image can be obtained by subtracting the residual map from the rainy image as illustrated in Fig. 2. Since the embedding guides residual map generation, according to the image representation theory in [16], we try to find a functional relationship between basic embedding features and the optimal ones in the latent mapping space. The error detector and the feature compensator are designed to address model errors, for approximating the optimal embedding features. The contributions of the paper are three-fold:

1) We analyzed the impact of uncertainty in the training process and the possibility of model error remapping via convolutional neural networks.

2) Based on the idea of closed-loop control, the error detector and the feature compensator are designed for addressing model error via feedback mechanism.

3) Unlike existing image deraining methods that embed low-quality features into the model directly, we replace low-quality features with latent high-quality features. This is a new perspective for improving the deraining performance.

2. Related Work

2.1. Single Image Deraining

Single image deraining plays a basic role for deraining. Unlike video based methods which analyse the difference between adjacent images, it is hard to remove rain streaks from a single image for lack of temporal information and ill-posed nature. For the more challenging single image deraining, traditional methods based on specific optimisition methods [10, 27, 3, 32, 14] often tend to produce degraded images due to the limited mapping transformation. Recently, with the aid of the CNN, [6] first focused on high-frequency rainy information for rain streak removal and demonstrated impressive restoration performance of the negative rain streak mapping network. Motivated by the deep residual neural network [13], [5] proposes a deep detail network that is also trained in high-frequency domain to reduce range of intermediate variables from input to output. To handle heavy rain streaks, Li et al. decomposes a rainy image into a background layer and several rainy layers at different depth, and a recurrent CNN is proposed to remove rain streaks at state-wisely different depths [17]. Similarly, In [23], a context aggregation network combined with memory units is used to remove rain streaks stage by stage.

2.2. Representation Learning

A high-quality residual map can be generated if conditional embedding features are able to depict contents of the residual map. Yang et al. [30] decomposed a rainy image into a background layer and a rain streak layer, and located rain streaks via the binary map. However, both of removed details and remaining rain streaks on the clean image reflected the limitations of simple binary map guidance. Different from [30], Zhang et al. [33] took the rain density into account to guide the network learning. Compared with the inadequacy of the rain density label that only represents image-level rain features, Qian et al. [22] proposed to use the attention map to guide the residual map generation. We interpret that effectiveness mainly comes from the introduction of the rain streak features that guides the network to focus more on the rain streak regions. Later, Wei et al. [28] introduced that the combined multi-stream convolutional structure can better describe context information. Despite the improved performance, these methods learned the image-level or pixel-level representation with-

out considering the detail loss caused by the uncertainty during training. Different from the structural improvement, a confidence guided method [31] studied the benefits from the residual map and its confidence. However, [31] simply used weight coefficients as confidence properties, which failed to consider the suitable confidence representation, error distribution complexity and error compensation. Thus, this method tends to miss details or introduce halo artifacts.

3. Feedback based Representation Learning

3.1. Problem Formulation

In [15, 4], the authors explained two types of uncertainty i.e. epistemic uncertainty a.k.a model uncertainty or systematic uncertainty, and aleatoric uncertainty a.k.a. statistical uncertainty that captures noise inherent in the observations. The variational inference can formulate epistemic uncertainty to compute variance. Maximum-aposterior or maximum-likelihood inference can formulate aleatoric uncertainty. To concisely describe our conditional optimization task, we model two output (conditional output and target output) and assume that probability distribution of each satisfies the Gaussian distribution. The minimisation objective, $\mathcal{P} = -\log p\left(y_1, y_2 | f^W(x)\right)$, is given as:

$$
\begin{aligned}
\mathcal{P} = &-\log p\big(y_1 \,|\, f^{W_1}\big(x, f^{W_2}(x)\big)\big) - \log p\left(y_2 \,|\, f^{W_2}(x)\right) \\
&\propto \frac{1}{2\sigma_1^2} \left\| y_1 - f^{W_1}\big(x, f^{W_2}(x)\big) \right\|^2 \\
&+ \frac{1}{2\sigma_2^2} \left\| y_2 - f^{W_2}(x) \right\|^2 + \log \sigma_1 \sigma_2.
\end{aligned}
\tag{1}
$$

where $p(\cdot)$ represents the probability function, $f^{W_i}(\cdot)(i = 1 \text{ or } 2)$ is the function of the corresponding network, x is the input rainy image and $y_i(i = 1 \text{ or } 2)$ is the output. We denote the mean of distribution $p\left(y_i | f^{W_i}(\cdot)\right) (i = 1 \text{ or } 2)$ as y_i and the variance as σ_i^2. The operations with subscript label 2 are introduced to generate the embedding residual map to depict the contents of the residual map truth. The operations with subscript label 1 are introduced to generate the final residual map with the aid of the embedding residual map, and the clean image can be obtained by subtracting the residual map from the rainy image. Due to the uncertainty during training, the effect of changing the various modules in the network is limited. Thus we improve the deraining performance based on finding a functional relationship between the basic embedding residual map and the optimal one in the latent mapping space.

Feedback mechanism. After model training, some embeddings with large errors often increase uncertainty to degrade the deraining performance [15]. To obtain high-quality features, the spirit of closed-loop feedback is incorporated into the CNN. In the automatic control system, the closed-loop control is capable of reducing the error through the integral
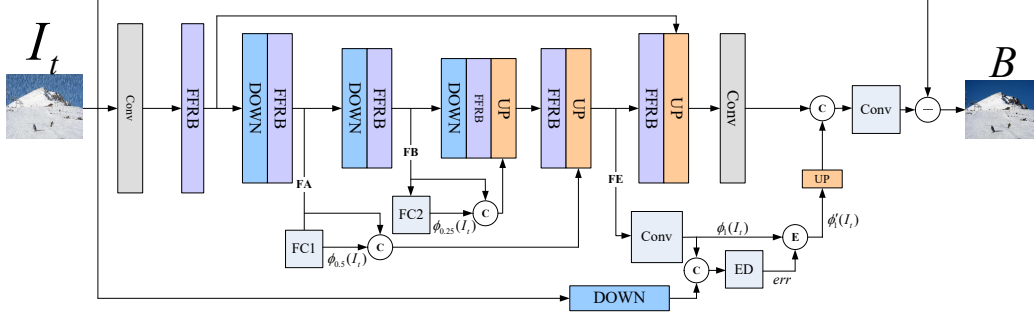
Figure 2. Compact overview of the proposed robust representation learning network structure (RLNet).
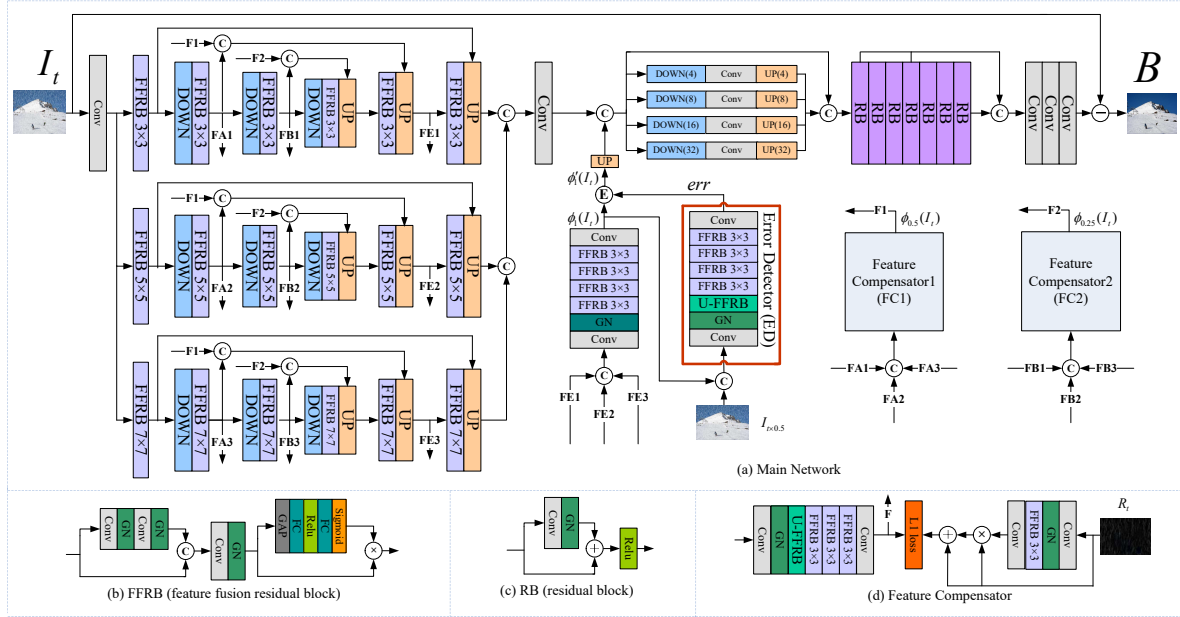

Figure 3. Full overview of the proposed robust representation learning network structure (RLNet). The ReLU functions after Conv and GN blocks are hidden for brevity. $-x \rightarrow$ represents information flow with index x, $\textcircled{E}$ is error compensation, $\textcircled{c}$ is concatenation operation, $\bigotimes$ is pixel-wise multiplication, $\bigoplus$ is pixel-wise addition and $\bigominus$ is pixel-wise subtraction. Zoom in to see small blocks better.

function and the feedback. To apply the idea of the closed-loop control, we treat the training datasets as discrete inputs. As shown in Fig. 2, after generating embedding residual map $\phi_1(I_t)$, we introduce the error detector (ED) with embedding residual map $\phi_1(I_t)$ and rainy image I_t as inputs to learn the error between embedding residual map $\phi_1(I_t)$ and corresponding residual map truth R_t. The resulting error map is used to compensate the embedding residual map for the better feature representation.

Effect analysis. The role of the error detector is reflected in the training process and the results. For the training process, the effect of error compensation on the embedding residual map enables the R-to-R module (connecting this embedding residual map and the final output residual map) to be learned more accurately. With the training convergence of the R-to-R module, the final output loss function have a smaller impact on the O-to-R module (connecting the input rainy image and this embedding residual map), so that O-

to-R module learns mainly based on the embedding feature loss function and acts as a feature guidance module. The decoupled learning is helpful to obtain appealing results [36]. To this end, the model error induced by uncertainty can be reduced by error compensation. For the results, the rectified embedding residual map are always more conducive to generate better deraining results as shown in Fig. 5.

3.2. CNN-based feedback process

Error distribution complexity. In general, the variable errors caused by the uncertainty during training can hardly be remapped by CNN due to the complexity of the error distribution. By abstracting error reciprocals as points that fluctuate above and below the zero value, Fig. 4 abstractly shows that operation of taking the absolute value of the error reciprocals can reduce the complexity of the error reciprocals distribution. Furthermore, the upper limit further simplifies the complexity of the error reciprocals distribution.

Small errors (corresponding to the large error reciprocals) that have little effect on the results are truncated by the upper limit, which is conducive to remap the error map.

Error detector. Specifically, we adopt the reciprocal of absolute errors multiplied by the threshold parameter θ_1 as training value to train the error detector, and use the sigmoid function at the end of the error detector. In such a case, the upper limit of the error detector output is 1, and the error detector output is closer to 0 as long as the error is larger. For the embedding residual map and the error detector, the corresponding optimization problems are:

$$\min_{\phi_1(\cdot)} \|R_{t\times 0.5} - \phi_1(I_t)\|_1, \qquad (2)$$

$$\min_{\varphi(\cdot)} \|\frac{\theta_1}{|R_{t\times 0.5} - \phi_1(I_t)|} - \varphi(I_{t\times 0.5}, \phi_1(I_t))\|_1, \qquad (3)$$

where I_t is the rainy image, $I_{t\times 0.5}$ is the rainy image at 0.5 scale size of I_t, R_t represents the residual map truth, $R_{t\times 0.5}$ is the residual map truth at 0.5 scale size of R_t, $\phi_1(\cdot)$ is the function trained to map I_t as the residual map $(\times 0.5)$, $\varphi(\cdot)$ is the function of the error detector, and θ_1 represents the threshold parameter. Due to the sparseness of rain streaks, reducing the size of the residual map by half can well represent the original residual map (see Fig. 5(d)(f)), and can greatly reduce the amount of calculation. Note that the last part of $\varphi(\cdot)$ is the sigmoid funtion and small errors are truncated by the upper limit. In addition, It should be emphasized that $|R_{t\times 0.5} - \phi_1(I_t)|$ in the optimization problem (3) is fixed, and the operation of taking the absolute value of the error reciprocal is very important for reducing the error distribution complexity. The absolute error map (named err in Eq. 4) and the rectified embedding residual map is calculated by the following rectification process:

$$err = \frac{\theta_1}{\varphi(I_{t\times 0.5}, \phi_1(I_t))} - \theta_1, \qquad (4)$$

$$\phi_1'(I_t) = \phi_1(I_t) - err(\mathbf{1} - 2\phi_1(I_t)). \qquad (5)$$

where $\mathbf{1}$ is all-one matrix and $\phi_1'(I_t)$ represents the rectified embedding residual map. In general, absolute errors (see Eq. 4) can not compensate features well. For this reason, we introduce a method to reasonably use absolute errors.

For the generated clean image, regions that should be rain-free always exist rain streaks due to heavy rain regions involved in the training process. Specifically, for the local optimal solution with locally similar inputs composed of rain and backgrounds, small pixel values and large pixel values of the residual map continuously fit the network to each of them during training, which indicates that smaller pixel values of the residual map are more likely to be generated by the network with larger values due to the pull of heavy rain pixels during training, especially when backgrounds of input rainy images are similar such that inputs
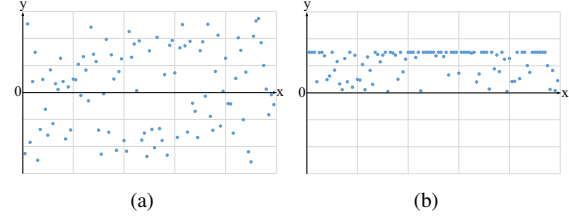

Figure 4. Abstract mapping from feature points (X-axis) to error reciprocals (Y-axis). (a) Distribution of 99 random error reciprocals within a certain range. (b) Distribution of random error reciprocals after taking the absolute value and upper limit truncation.

for the network convolution are similar. Hence, the embedding residual map $\phi_1(I_t), (0 \leq \phi_1(I_t) \leq \mathbf{1})$ can describe the trend of corresponding errors. Note that the magnitude of embedding residual map values is much larger than error values. We multiply the absolute error map by the coefficient $(\mathbf{1} - 2\phi_1(I_t))$ that represents the confidence map of the absolute error map. To this end, the transformed error map (see Fig. 5(e)) are used to compensate the embedding residual map as shown in Eq. 5. The embedding residual map value from small to large corresponds to the error map value from negative to positive in terms of probability.
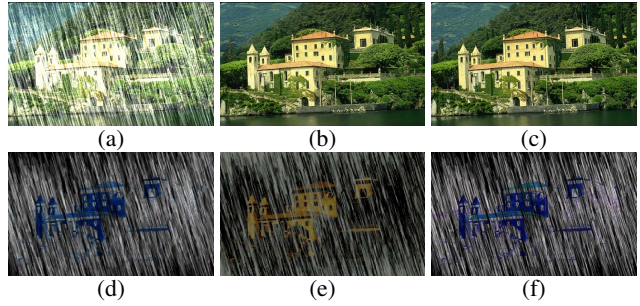

Figure 5. (a) Rainy image. (b) Derained result of RLNet. (c) Groundtruth. (d) Embedding residual map $(\times 0.5)$ not recrified by the error detector. (e) Error map multiplied by 10 for visualization. (f) Final generated residual map.

Feature compensator. Similarly, our goal in the feature compensator is to design a light-weight module to obtain the high-quality embedding residual map to guide other embedding residual map. In this case, the embedding residual map that needs to be rectified by the error detector becomes more accurate to reduce the difficulty of error remapping. According to Eq. 1, we wish to obtain a embedding residual map less affected by uncertainty. Recalling that the CNN is of the complex nonlinear function, we adopt the CNN-based module (see Fig. 3(d)) to transform residual map truth R_t for error compensation, which can be learned as follows:

$$\min_{\phi_i(\cdot),\omega_i(\cdot)} \|R_{t\times i} + \theta_2\omega_i(R_{t\times i})R_{t\times i} - \phi_i(I_t)\|_1 + \zeta, \quad (6)$$

where $\zeta = P(\omega_i(R_{t\times i})) = \lambda \times \|\omega_i(R_{t\times i})\|_2^2$, $P(\cdot)$ means the regularizer, λ is the setup parameter, $R_{t\times i}$ represents the

residual map at i ($i = 0.25$ or 0.5) scale size of R_t. $\phi_i(\cdot)$ is the function trained to map I_t as the transformed residual map at i scale size (as shown in Fig. 6(d)(e)), $\omega_i(\cdot)$ represents the transformation function and θ_2 is the transformation parameter that determines the level of transformation. As $\omega_i(\cdot)$ constrains the error by transforming features, the loss of details caused by uncertainty will be reduced.
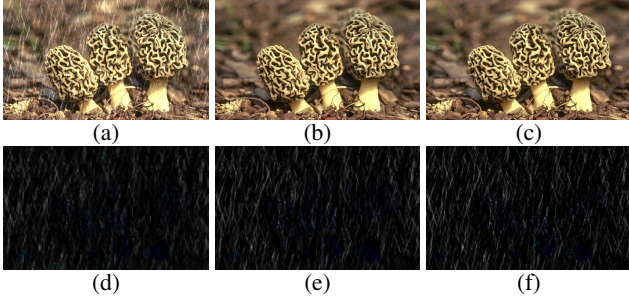

Figure 6. (a) Rainy image. (b) Derained result of RLNet. (c) Groundtruth. (d) Embedding residual map ($\times 0.25$) output by the feature compensator. (e) Embedding residual map ($\times 0.5$) output by the feature compensator. (f) Final generated residual map.

3.3. Network Architecture

Feature Fusion Residual Block. We adopt the group normalization [29] and the SE block [17] for the better feature fusion. As shown in Fig 3(b), the feature fusion residual block (FFRB) is formulated as:

$$FFRB(x) = SE(GN(\text{Conv}(\text{Res}(x)))), \qquad (7)$$

where x is the input signal and $\text{Res}(\cdot)$ represents residual block containing group normalization (GN). Since the better performance of group normalization than batch normalization and instance normalization when the batch size is small [29], the group normalization is used to reduce internal covariate shift. By using SE block [17], the feature channels with more contextual information will be intensified by the larger weight coefficient offered by SE block. Mathematically, SE block can be expressed as:

$$S(x) = \text{Sigmoid}(FC(\text{Relu}(FC(GAP(x))))) \otimes x, \quad (8)$$

where $GAP(\cdot)$ is the global average pooling and $FC(\cdot)$ is the fully connected layer.

Multi-stream Residual Architecture. The proposed deraining network is built upon the encoder-decoder architecture that is widely adopted in image deraining [24, 31]. A large receptive field induced by encoder-decoder module U-FFRB is able to obtain context information. Since rain streaks commonly at defferent shape, density and scale, only utilizing one-stream CNN [23, 24, 31] may lose some useful information. Motivated by the effectiveness of multi-scale convolutional kernels [33, 28], we use the multi-stream residual network to extract multi-scale concentrated features. The kernel sizes of the FFRB belonging to their respective streams are set to 3, 5, and 7, respectively, to find back the lost details. From Fig 3(a), the multi-stream architecture can be described as:

$$M(x) = [U_{3\times3}(x), U_{5\times5}(x), U_{7\times7}(x)], \qquad (9)$$

where $U_{i\times i}(\cdot)$ represents U-FFRB module with the kernel size i, and $[\cdot]$ is the concatenation operation.

Error detector and feature compensator. For the feature compensator, an additional feature transformation module is introduced to adaptively transform residual map truth, and a concise encoder-decoder branch is constructed as shown in Fig 3(d) to learn the transformed residual map. For the correlation between different streams, the learned transformed residual map is copied into three copies and respectively embedded into three streams with skip-connection as shown in Fig 3(a). For the error detector, two concise branches containing FFRB are constructed to learn the residual map and the error map. Noted that one of the branches constructed for error map generation contains the encoder-decoder module. The error map is used to compensate corresponding embedding residual map (see Fig 3(a)).

Refinement module. The main goal of the refinement module is to finely adjust the feature maps clustered together. We use the spatial pyramid pooling [9] to further obtain multi-scale features. The scale factors are set to 4, 8, 16 and 32, respectively. For the feature maps with different sizes, the point-wise convolution is utilized to reduce their channels and the up-sample operation adopting the nearest interpolation is utilized to restore original size. As shown in Fig 3(a), the down-up structure can be formulated as:

$$D(x) = [(\text{Conv}(x \downarrow_4)) \uparrow_4, (\text{Conv}(x \downarrow_8)) \uparrow_8, \\ (\text{Conv}(x \downarrow_{16})) \uparrow_{16}, (\text{Conv}(x \downarrow_{32})) \uparrow_{32}, x]. \qquad (10)$$

The next seven resblocks with the group normalization [29] are designed as shown in Fig 3(c).

3.4. Loss Function

The squared penalty of Mean Square Error (MSE) penalizes large errors and tolerates small errors, which tends to produce the over-smoothed image. Thus, Mean Absolute Error (MAE) is used to obtain better robustness. For the error detector rectifying the embedding residual map, we minimize the following two loss functions:

$$L_{e1} = \|R_{t\times0.5} - \phi_1(I_t)\|_1, \qquad (11)$$

$$L_{e2} = \left\| \frac{\theta_1}{|R_{t\times0.5} - \phi_1(I_t)|} - \varphi(I_{t\times0.5}, \phi_1(I_t)) \right\|_1, \quad (12)$$

Note that $|R_{t\times0.5} - \phi_1(I_t)|$ in Eq. 12 is fixed. For the feature compensator, the loss function and the regularizer
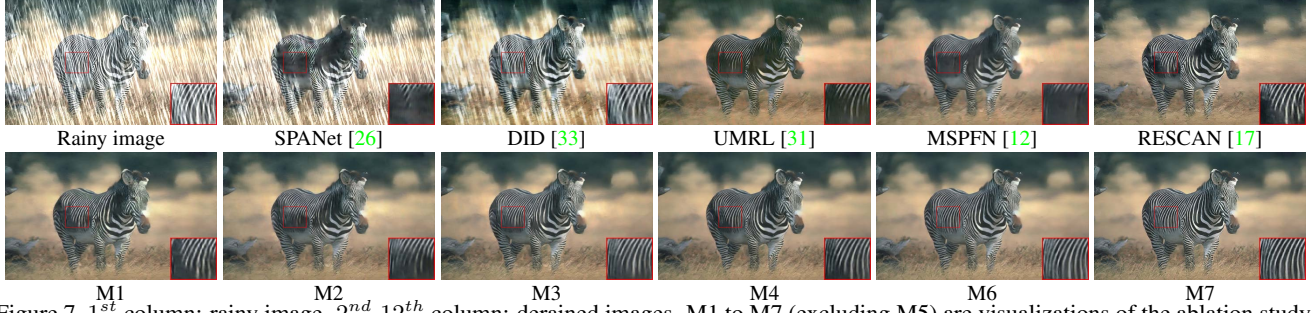
Figure 7. 1st column: rainy image. 2nd-12th column: derained images. M1 to M7 (excluding M5) are visualizations of the ablation study.

are formulated as follow:

$$L_c = \sum_{i \in \{0.25, 0.5\}} \|R_{t \times i} + \theta_2 \omega_i (R_{t \times i}) R_{t \times i} - \phi_i (I_t)\|_1,$$
(13)

$$L_p = \|\omega_i (R_{t \times i})\|_2^2,$$
(14)

In the proposed model, the final output residual map is refined via using the following loss function:

$$L_f = \|R_t - f(I_t)\|_1,$$
(15)

where $f(\cdot)$ represents the overall network function. Furthermore, motivated by the goal of generating derained image to approximate its ground-truth image, we utilize the SSIM function [37] as the additional evaluation metric for the generated clean image. It is formulated as follows:

$$L_{SSIM} = -SSIM(B_t, I_t - f(I_t)),$$
(16)

where B_t is the clean image truth. The combination of the MAE based loss function and SSIM loss function can preserve the per-pixel similarity as well as preserving the global structure. The overall loss function used to train proposed RLNet is formulated as follows:

$$L_{all} = L_f + L_{SSIM} + \lambda L_p + \lambda_1 L_{e1} + \lambda_2 L_{e2} + \lambda_3 L_c, \quad (17)$$

where λ, λ_1, λ_2 and λ_3 are setup parameters.

4. Experiments and Results

4.1. Experiment Settings

Datasets. For fairness, we use the same benchmark datasets as other methods. **DID-data** with different rain magnitudes provided by Zhang et al. [33] contains 12000 training images and 1200 testing images, **Rain800** collected by Zhang et al. [34] contains 700 training images and 100 testing images. Two datasets are synthesized by Yang et al. [30], namely **Rain200H** and **Rain 100L**. Rain200H (with heavy rain streaks) consists of 1800 training images and 200 testing images, and Rain100L (with light rain streaks) consists of 200 training images and 100 testing images. In addition, [23, 26] supply some real-world rainy images as a test set. Deraining methods are trained on the synthetic dataset Rain200H but are tested on real-world images.

Training Details. During training, a image pair (or its horizontal flip) is resized as 512×512 for training. We implemented all experiments on a Titan Xp GPU and use a batch size of 1. The first stage is used to train the error detector and the feature compensator with the initial value $\lambda = 0.01$, $\lambda_1 = 0.6$, $\lambda_2 = 0$, $\lambda_3 = 0.6$, $\theta_1 = 0$, $\theta_2 = 0.05$: Adam is used as the optimizer to train the model and ends after 90 epochs. The initial learning rate is 0.0002 and divided by 5 when reaching 50, 65, 80 epochs. When reaching 20 epochs, θ_2 is set as 0.15. When reaching 30 epochs, λ_2 is set as 6. By using the pre-trained weights obtained in the first stage, the fine-turning training process with the initial value $\lambda = 0.01$, $\lambda_1 = 0.6$, $\lambda_2 = 0$, $\lambda_3 = 0.6$, $\theta_1 = 0.15$, $\theta_2 = 0.05$ is as follows: The initial learning rate is 0.0002 and divided by 2 every 30 epochs. The 240 epochs in total. When reaching $30 \times K (K = 1, 2, 3, 4, 5, 6)$ epochs, λ_2 is set as 0. When reaching $30 \times K + 15 (K = 0, 1, 2, 3, 4, 5)$ epochs, λ_2 is set as 0.6.

Quality Comparisons. Peak Signal to Noise Ratio (PSNR) [11] and Structural Similarity Index (SSIM) [37] are adopted to assess the performance of different methods on benchmark datasets. Since there is no ground-truth image in real-world datasets, NIQE [21] is adopted to evaluate the generalization ability of different methods on real datasets.

4.2. Ablation Study

We conduct all ablation experiments on Rain200H for its heavy rain streaks distributed heterogeneously.

Absolute value operation. Using the non-absolute error map to directly add to the embedding residual map, a network termed as RLNet- with simple error compensation is constructed. Since the error map as a variable target is difficult to fit, even if we double the parameters of the error detector for RLNet-, the resulting 27.70 dB only exceeding the original size model RLNet- in Table 2 by 0.13 dB.

Threshold parameter θ_1. In order to obtain a light-weight error detector, the absolute value operation is adopted and the threshold parameter θ_1 is introduced to dynamically adjust upper limit of error reciprocals for better error map remapping as shown in Eq. 12. We keep a certain $\theta_2 = 0.15$

Table 1. PSNR and SSIM comparisons on four benchmark datasets. Red and blue colors are used to indicate 1st and 2nd rank, respectively. ▷ denotes some metrics of this method are copied from [19]. ° denotes the model is re-trained due to no pre-trained wight provided by the author. To be fair, UMRL [31] and MSPFN [12] are fine-tuned with the Rain100L training set when evaluated on the Rain100L test set.

Method	LP [18]	DSC [20]▷	DDN [6]	DID [33]	SPANet [26]	RESCAN [17]°	UMRL [31]	PReNet [23]	MSPFN [12]	RCDNet [25]	RLNet
DID-data [33]	22.46/0.801	21.44/0.789	27.33/0.853	27.93/0.861	22.96/0.720	29.12/0.880	30.35/0.891	30.40/0.891	30.34/0.881	29.81/0.859	32.62/0.917
Rain200H [30]	14.26/0.420	15.66/0.544	20.12/0.635	15.54/0.520	13.27/0.412	25.92/0.823	23.01/0.744	27.64/0.884	24.30/0.748	28.83/0.886	28.87/0.895
Rain100L [30]	29.11/0.881	24.16/0.866	33.50/0.944	23.79/0.773	27.85/0.881	36.58/0.970	32.39/0.921	36.28/0.979	33.50/0.948	38.60/0.983	37.38/0.980
Rain800 [34]	20.46/0.729	18.56/0.599	21.16/0.732	21.22/0.750	21.22/0.687	23.90/0.828	23.24/0.808	22.83/0.790	25.52/0.830	24.59/0.821	27.95/0.870

Table 2. Effect of threshold parameter θ_1.

Method	RLNet-	$\theta_1 = 0.03$	$\theta_1 = 0.04$	$\theta_1 = 0.05$	$\theta_1 = 0.06$
PSNR	27.57	28.48	28.57	**28.87**	28.61
SSIM	0.856	0.877	0.882	**0.895**	0.881

Table 3. Effect of the error detector. $-E$ means that RLNet does not contain the error detector. $\pm E$ denotes that RLNet uses the error detector in training but removes it after training. RLNet is our default model containing the error detector.

Method	UMRL*[31]	RLNet$-E$	RLNet$\pm E$	UMRL*$+E$	RLNet
PSNR	22.31	27.51	28.05	24.45	**28.87**
SSIM	0.767	0.851	0.880	0.791	**0.895**

Table 4. Effect of transformation parameter θ_2.

Method	$\theta_2 = 0.05$	$\theta_2 = 0.1$	$\theta_2 = 0.15$	$\theta_2 = 0.2$
PSNR	28.74	28.81	**28.87**	28.85
SSIM	0.889	0.890	**0.895**	0.893

Table 5. Ablation study on different modules.

Model	M1	M2	M3	M4	M5	M6	M7
Base Network	✓	✓	✓	✓	✓	✓	✓
FFRB		✓	✓	✓	✓	✓	✓
Multi-stream			✓	✓	✓	✓	✓
Embedding				✓	✓	✓	✓
E-Detector					✓	✓	✓
L_{e2}						✓	✓
$L_c + L_p$							✓
PSNR	26.91	27.13	27.42	27.39	27.40	28.69	**28.87**
SSIM	0.830	0.833	0.847	0.846	0.847	0.881	**0.895**

fixed and change the θ_1 to find a better parameter setting for the error detector. Table 2 lists the PSNR and SSIM values of four RLNet models with $\theta_1 = 0.03, 0.04, 0.05, 0.06$. When $\theta_1 = 0.03, 0.04, 0.06$, the RLNet performs a little inferior to RLNet with $\theta_1 = 0.05$. It can be interpreted as the θ_1 that is too large will weaken the role of the error detector, and the θ_1 that is too small will increase the difficulty of error map remapping. Then we set the better parameter setting ($\theta_1 = 0.05$) for the proposed RLNet.

The results in Table 3 verify the effect of the error detector in the training process and results. In addition, we apply the embedding residual map and the error detector to the base network UMRL* [31]. The obtained UMRL*$+E$ achieves 2.14 dB performance improvement.

Transformation Parameter θ_2. We keep a certain $\theta_1 = 0.05$ fixed and change the θ_2 to find a better parameter setting for the feature compensator. Table 4 lists the PSNR and SSIM values of four RLNet models with $\theta_2 = 0.05, 0.1, 0.15, 0.2$. RLNet with $\theta_2 = 0.05, 0.1, 0.2$ performs a little inferior to RLNet with $\theta_2 = 0.15$. For the too small value of θ_2 weakening the role of the feature compensator and the too large value of θ_2 increasing the complexity of feature distribution, it is resonable to see those results from Table 4. Hence, we set $\theta_2 = 0.15$ for the RLNet.

Network Architecture. As shown in the Table 5), M1 denotes a single-stream U-net (kernel size is 3) without other modules. The base network, M1 itself enables the PSNR and SSIM to reach 26.91 dB and 0.830% respectively. M2 replaces the residual block with FFRB for M1. The FFRB increases the PSNR by 0.22 dB and the SSIM by 0.3%. Moreover, M3 modifies single-stream mechanism of M2 to multi-stream mechanism, with which M3 surpasses M2 by 0.29 dB and 1.4%. When compared with M3, M4 with the unrectified embedding residual map hardly improves performance. M6 adds the error detector to M4. By comparing M6 and M4, it can be seen that error detector by itself con-

tributes 1.29 dB and 3.4%. M5 in Table 5) is obtained from M6 by removing the error detection oriented loss function L_{e2} but keeping all other conditions unchanged. Experimental results show that M5 no longer achieves the same performance as M6 but has degraded performance as M4. This implies that the proposed error detection mechanism does play an important role in enhancing the model performance. Besides, the RLNet$\pm E$ in Table 3 that uses the error detector in training but removes it after training will finally have the same model size as the RLNet without using error detection at all, yet the former outperforms the latter. M7 adds $L_c + L_p$ to M6 (i.e. θ_2 is changed from 0 to 0.15). The results show that M7 with the feature compensation obtains the better performance over M6 by 0.18 dB and 1.4%. The LPIPS [35] values of M4 (without error detector), M5 (without L_{e2}) and M6 (with error detector) are 0.053, 0.052 and 0.041 respectively when tested on Rain200H. From Fig. 7, one can see that the error detector and the feature compensator further improve deraining performance.

4.3. Experiments on Benchmark Datasets

Ten widely adopted methods are compared with our network, including two traditional methods, i.e., **DSC** [20] and **LP** [18], and eight state-of-the-art deep learning based methods, i.e., **DDN** [6], **DID** [33], **RESCAN** [17], **SPANet** [26], **UMRL** [31], **PReNet** [23], **MSPFN** [12] and **RCDNet** [25]. Results of our method and other methods are given in Table 1. We select seven images from four benchmark datasets to visually validate the qualitative performance of different methods. Specifically, visually one can see in the second and third columns of Fig. 10, though the light rain streaks can be removed by SPANet [26] and DID [33], remaining rain streaks with missing image de-

(a) Rainy images   (b) SPANet [26]   (c) RESCAN [17]   (d) DID [33]   (e) UMRL [31]   (f) MSPFN [12]   (g) Ours   (h) Groundtruth

Figure 8. Examples of joint deraining and segmentation. DeepLabv3+ [2] is adopted for segmentation. Zoom in to see the details.
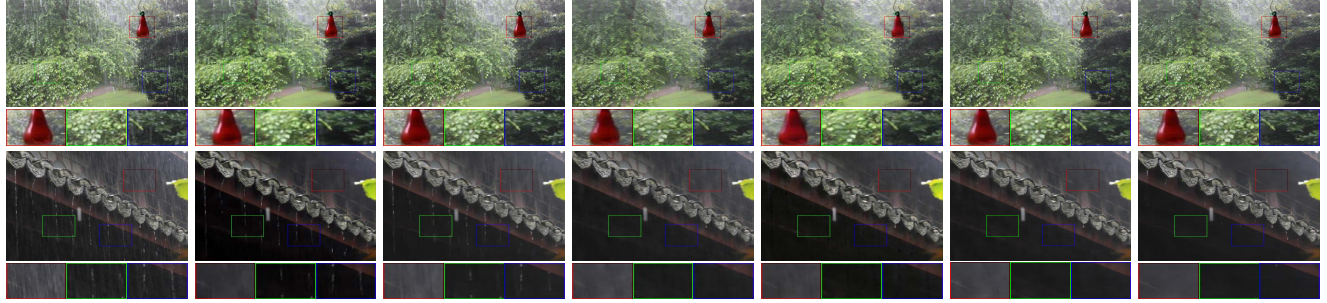
(a) Rainy images   (b) DID [33]   (c) MSPFN [12]   (d) RESCAN [17]   (e) UMRL [31]   (f) RCDNet [25]   (g) Ours

Figure 9. Visual quality comparisons on sample images from real-world datasets. Zoom in to see the details.
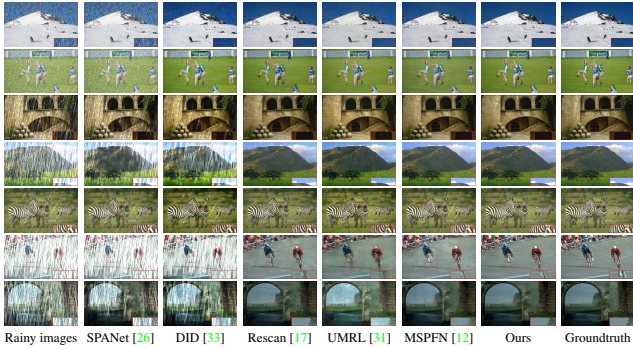
Rainy images   SPANet [26]   DID [33]   Rescan [17]   UMRL [31]   MSPFN [12]   Ours   Groundtruth

Figure 10. Visual quality comparisons on sample images from benchmark datasets. Zoom in to see the details.

tails usually exist when the rain is heavy. By comparing fourth, sixth and seventh rows of Fig. 10, the UMRL [31], RESCAN [17] and MSPFN [12] are hard to detect heterogeneously distributed rain streaks in different regions and recover clean image details well. We also test the derained image by DeepLabv3+ [2] as shown in Fig. 8. In addition, we randomly select 400 image pairs (512×512) to evaluate average time and performance of different methods as shown in Table 6. The error detector that takes little running time brings the considerable performance improvement. The FLOPs (in $\times 10^{11}$) of RESCAN, MSPFN, RCD-Net, RLNet±E, and RLNet are 1.3, 24.2, 7.8, 1.6 and 2.0.

4.4. Experiments on Real Rainy Images

Using the real-world datasets [26] cropped to 481×321, we compare proposed RLNet with other methods as shown in Table. 7. In addition, from Fig. 9, it can be observed that RLNet, UMRL [31] and RESCAN [17] outperform DID [33] and MSPFN [12] in removing heavy rain streaks from

Table 6. Time complexity (in seconds) and performance of different methods. Sizes of testing images are 512×512. RLNet−E means that RLNet does not contain the error detector.

Method	DSC	UMRL	RCDNet	RESCAN	MSPFN	RLNet−E	RLNet
PSNR	21.90	28.71	28.99	27.31	28.66	30.19	31.34
Avg time	371.13	8.831	3.023	0.952	1.040	0.333	0.373

Table 7. NIQE comparisons on real-world datasets, smaller scores indicate better image quality.

Method	SPANet	DID	MSPFN	UMRL	RCDNet	RESCAN	RLNet
NIQE↓	5.109	5.068	5.561	5.257	4.701	4.631	4.498

real rainy image. Specifically, as can be seen from the Fig. 9, there are more or less visible rain streaks in the results by DID [33] and MSPFN [12], while RLNet, UMRL [31] and RESCAN [17] can generate the clean image. For the fourth, fifth and sixth columns of the Fig. 9, the blurs and halo artifacts with missing image details exist in the results by UMRL [31] and RESCAN [17], while the rain streak removal results by RLNet are high-quality and clear.

5. Conclusion

We have proposed the RLNet for single image deraining. Based on the CNN in the presence of uncertainty, a new method for error detection and feature compensation is proposed for latent high-quality representation learning. An iterative optimization scheme that unrolls image optimization and error optimization with image priors and correction terms is presented. Experiments demonstrate that the proposed RLNet is robust enough to detect heterogeneous rain streaks and recover details for real rainy images. Taking the portability into account, the proposed method for error detection and feature compensation can be selectively incorporated into learning based image deraining networks.

References

[1] D. Chen, C. Chen, and L. Kang. Visual depth guided color image rain streaks removal using sparse coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(8):1430–1455, 2014.

[2] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

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

[4] R. Cipolla, Y. Gal, and A. Kendall. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7482–7491, 2018.

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

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

[7] N. Hautiere, J. Tarel, and D. Aubert. Towards fog-free in-vehicle vision systems through contrast restoration. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[8] K. He, J. Sun, and X. Tang. Single image haze removal using dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12):2341–2353, 2011.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015.

[10] D. Huang, L. Kang, Y. F. Wang, and C. Lin. Self-learning based image decomposition with applications to single image denoising. *IEEE Transactions on Multimedia*, 16(1):83–93, 2014.

[11] Q. Huynh-Thu and M. Ghanbari. Scope of validity of psnr in image/video quality assessment. *Electronics Letters*, 44(13):800–801, 2008.

[12] K. Jiang, Z. Wang, Y. Peng, C. Chen, B. Huang, Y. Luo, J. Ma, and J. Jiang. Multi-scale progressive fusion network for single image deraining. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8346–8355, 2020.

[13] He K, Zhang X, Ren S, and Sun J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2016.

[14] L. Kang, C. Lin, and Y. Fu. Automatic single-image-based rain streaks removal via image decomposition. *IEEE Transactions on Image Processing*, 21(4):1742–1755, 2012.

[15] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*, pages 5574–5584. 2017.

[16] K. Lenc and A. Vedaldi. Understanding image representations by measuring their equivariance and equivalence. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 991–999, 2015.

[17] X. Li, J. Wu, Z. Lin, H. Liu, and H. Zha. Recurrent squeeze-and-excitation context aggregation net for single image deraining. In *The European Conference on Computer Vision*, pages 262–277, 2018.

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

[19] X. Liang, B. Qiu, Z. Su, C. Gao, X. Shi, and R.Wang. Rain wiper: An incremental randomly wired network for single image deraining. *Computer Graphics Forum*, 38(7):159–169, 2019.

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

[21] A. Mittal, R. Soundararajan, and A. C. Bovik. Making a completely blind image quality analyzer. *IEEE Signal Processing Letters*, 20(3):209–212, 2013.

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

[23] D. Ren, W. Zuo, Q. Hu, P. Zhu, and D. Meng. Progressive image deraining networks: A better and simpler baseline. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3932–3941, 2019.

[24] G. Wang, C. Sun, and A. Sowmya. Erl-net: Entangled representation learning for single image de-raining. In *IEEE International Conference on Computer Vision*, pages 5643–5651, 2019.

[25] H. Wang, Q. Xie, Q. Zhao, and D. Meng. A model-driven deep neural network for single image rain removal. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3100–3109, 2020.

[26] T. Wang, X. Yang, K. Xu, S. Chen, Q. Zhang, and R. W. H. Lau. Spatial attentive single-image deraining with a high quality real rain dataset. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 12262–12271, 2019.

[27] Y. Wang, S. Liu, C. Chen, and B. Zeng. A hierarchical approach for rain or snow removing in a single color image. *IEEE Transactions on Image Processing*, 26(8):3936–3950, 2017.

[28] Y. Wei, Z. Zhang, H. Zhang, R. Hong, and M. Wang. A coarse-to-fine multi-stream hybrid deraining network for single image deraining. In *IEEE International Conference on Data Mining*, pages 628–637, 2019.

[29] Y. Wu and K. He. Group normalization. In *The European Conference on Computer Vision*, pages 3–19, 2018.

[30] W. Yang, R. T. Tan, J. Feng, J. Liu, Z. Guo, and S. Yan. Deep joint rain detection and removal from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1685–1694, 2017.

[31] R. Yasarla and V. M. Patel. Uncertainty guided multi-scale residual learning-using a cycle spinning cnn for single image

de-raining. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8397–8406, 2019.

[32] H. Zhang and V. M. Patel. Convolutional sparse and low-rank coding-based rain streak removal. In *IEEE Winter Conference on Applications of Computer Vision*, pages 1259–1267, 2017.

[33] H. Zhang and V. M. Patel. Density-aware single image de-raining using a multi-stream dense network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 695–704, 2018.

[34] 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*, pages 1–1, 2019.

[35] Richard Zhang, I. Phillip, E. A. Alexei, S. Eli, and W. Oliver. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[36] Z. Zhang, Y. Song, and H. Qi. Decoupled learning for conditional adversarial networks. In *IEEE Winter Conference on Applications of Computer Vision*, pages 700–708, 2018.

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